Electrónica e Telecomunicações

universidade de aveiro

AVEIRO • MAR • 2005 • VOL. 4 • Nº4
Revista do Departamento de Electrónica e Telecomunicações da Universidade de Aveiro

Electrónica e Telecomunicações

Revista do Departamento de
Electrónica e Telecomunicações
da Universidade de Aveiro

Morada e Secretariado:

Departamento de Electrónica
e Telecomunicações
Universidade de Aveiro
Campo Universitário
3810 AVEIRO
Portugal

Artes Gráficas:

Sérgio Cabaço

Produção: DESIGNEED, Lda
Tiragem : 300 exemplares
Depósito Legal Nº 115607

ISSN: 1645-04

Editorial

Este número da Revista do DET continua a evidenciar a habitual diversidade de interesses científicos e pedagógicos que determinam a nossa actividade como membros do DET. Basta uma rápida análise do índice para identificar temas das áreas das telecomunicações, sistemas de informação, processamento de sinal e imagem e sistemas computacionais, e controlo.

Num momento em que a cultura do *online* está na ordem do dia é inevitável equacionar o interesse e a apetência que uma iniciativa editorial como esta possa continuar a suscitar por parte dos seus mais directos colaboradores: o corpo docente do DET e uma significativa parte do corpo discente do DET com ênfase natural para aqueles em fase final de graduação ou pós-graduação. A ideia que pessoalmente aqui gostaria de veicular, fundada pela experiência de alguns anos de actividade editorial, é a de que é superior o grau de realização de quem vê o seu trabalho publicado desta forma que, de facto, se pode considerar tradicional. É evidente que, a breve trecho, teremos uma versão *online*, da revista mas, outra vez em registo pessoal, a impressão de algo efémero é muito mais atenuada com a publicação no formato que nos vem habituando há já onze anos.

Aguardando desde já contribuições para uma próxima edição, apresento o meu agradecimento aos autores e a todas as entidades que duma ou doutra forma viabilizaram a concretização de mais uma edição desta Revista.

Augusto Silva

A edição desta revista é subsidiada pela
Fundação para a Ciência e Tecnologia

Índice

Extracção de Informação de Relatórios Médicos

Liliana Ferreira, António Teixeira, João Paulo Cunha

Abstract – **This paper presents the first steps given to develop an information extraction system for portuguese texts. The system intends the extraction of information from medical reports and is based on the GATE system developed in the University of Sheffield. We present the changes made to this system in order to adapt it to the information extraction in Portuguese and some examples of results already gotten.**

Resumo – **Neste artigo apresentam-se os primeiros passos dados no sentido de desenvolver um protótipo de um sistema para a extracção de informação em Português. O sistema tem como domínio de aplicação relatórios médicos da área da neurofisiologia e baseia-se no sistema GATE desenvolvido na Universidade de Sheffield. As alterações efectuadas a este sistema com o intuito de o adaptar à extracção de informação em português são apresentadas, bem como alguns exemplos de resultados já obtidos.**

Palavras chave – **Extracção de Informação, Processamento de Linguagem Natural, Recuperação de Informação, *Text Mining***

I. Introdução

A Extracção de Informação é um dos campos de uma área de estudos mais abrangente: o Processamento de Linguagem Natural, que estuda os idiomas humanos a partir de uma perspectiva computacional. Outros campos desta área são, por exemplo, a Extracção de Conhecimento de Texto (*Text Mining*), que tem como objectivo a descoberta, reconhecimento e derivação de nova informação a partir de um grande conjunto de textos, e a Recuperação de Informação (*Information Retrieval*), cujo objectivo passa pela obtenção de informação relevante a partir de um amplo conjunto de textos, sendo as suas técnicas tipicamente utilizadas para obter documentos relevantes a partir de um conjunto de vários tipos de documentos, entre outras.

A extracção de informação (doravante IE do inglês *Information Extraction*) automática de textos envolve decidir se um texto é relevante para um dado domínio e caso seja, extrair um conjunto de factos desse texto. No entanto, a maior parte dos sistemas de IE foram desenvolvidos para textos escritos na língua inglesa. Actualmente, a IE em inglês está muito próxima do desempenho de especialistas humanos.

Um tipo de informação que abunda nos ambientes hospitalares é a informação escrita, cada vez mais em formato digital, e a informação transmitida oralmente entre vários intervenientes nos processos clínicos. Num cenário em que vingue a utilização sistemática de sistemas de transcrição de relatórios de uma forma automática, será necessário que os sistemas tenham cada vez mais capacidades de associar significado às palavras e frases com que lidam. Mesmo noutros ambientes, como a web, assiste-se a evoluções no sentido de uma *semantic web* em que a informação terá *tags* facilitando a procura por conceitos e não pelas actuais palavras.

Deste modo, o desenvolvimento de sistemas que obtenham informação existente em relatórios médicos de uma forma automática tornaria acessível a grande quantidade de informação deste tipo existente em ambientes hospitalares.

II. Extracção de Informação

A IE é uma tecnologia que transforma dados não-estruturados de documentos, em informações explícitas; isola partes relevantes do texto, extrai informação dessas partes e transforma-as em informações mais digeridas e melhor analisadas. Além disto, permite também formatar as informações recolhidas nos textos construindo padrões de saída (por exemplo, bancos de dados estruturados ou frases em linguagem natural).

O objectivo da IE direcciona-se para a necessidade de recolher informação produzindo dados estruturados a partir de um número indefinido de textos, permitindo o desenvolvimento de processos de base de conhecimento fundamentado, ou seja, modelos específicos de ocorrências, entidades ou relações.

No entanto, do ponto de vista do processamento de linguagem natural (NLP do inglês *Natural Language Processing*), a IE é atractiva pois, as suas tarefas estão bem definidas. Para além disto, a IE utiliza textos reais e coloca problemas de NLP difíceis e interessantes. A *performance* da IE pode ser comparada à *performance* humana na mesma tarefa.

A. *Dificuldades que se colocam à IE*

A.1 *Portabilidade*

Uma das barreiras que se apresentam à IE é o custo de adaptar um sistema de extracção a um novo "cenário". Em geral cada aplicação de IE envolve um cenário diferente e se a implementação deste novo cenário exigir meses de trabalho e de intervenção dos *designers* do sistema, o mercado permanecerá limitado.

São, assim, necessárias ferramentas que permitam aos potenciais utilizadores adaptar e criar um sistema inicial em dias ou semanas e não em meses.

A questão básica que se põe ao desenvolvimento de tal ferramenta é a forma e o nível de informação desejado pelo utilizador. Uma das possibilidades é produzir uma representação gráfica dos modelos, mas esta solução expõem muitos detalhes dos modelos. Em vez disso, muitos grupos estão a desenvolver sistemas que obtêm informação principalmente de exemplos de frases de interesse e de informação a ser extraída.

A.2 *Desempenho*

Uma outra barreira à proliferação do uso de sistemas de extracção é a limitação do desempenho. O que poderá contribuir para o aumento do desempenho? Em parte a convergência de tecnologias: os melhores sistemas existentes actualmente são praticamente semelhantes na sua apresentação global. Por outro lado estão as características do domínio. A experiência de outros fenómenos linguísticos parece indicar que uma grande fracção de dados relevantes estão codificados linguisticamente por um pequeno número de formas. É ainda de notar que o aumento da investigação nesta área provoca o aumento de cenários de extracção implementados. Assim, pode-se esperar ver conjuntos de modelos que são aplicados a famílias de cenários relacionados ou a domínios gerais. Por exemplo, modelos para acções básicas como compra e venda de produtos podem ser aplicados a muitos cenários dentro do domínio do negócio.

III. *Message Understanding Conferences*

Durante uma década[1] a IE foi conduzida por conferências para a compreensão de mensagens (MUC[2]). Estas conferências, instituidas pela *Defense Advanced Research Projects Agency* (DARPA) do ministério da Defesa dos Estados Unidos da América, ajudaram a formalizar a IE. Como exemplo, as tarefas de IE, denominadas *MUC tasks*, eram especificadas e incluíam a avaliação de critérios e *corpora* de texto para teste. As *MUC tasks* são ainda amplamente utilizadas para a avaliação dos sistemas de IE. Estas tarefas estão resumidas na tabela I.

Actualmente várias conferências da área da linguística computacional e da inteligência artificial lidam com a IE e com as suas sub-tarefas.

MUC	MUC *task*
MUC-1 (1987) e MUC-2 (1989)	mensagens sobre operações navais
MUC-3 (1991) e MUC-4 (1992)	artigos noticiosos sobre actividades terroristas
MUC-5 (1993)	artigos noticiosos sobre parcerias e microelectrónica
MUC-6 (1995)	artigos sobre mudanças de gerência
MUC-7 (1997)	noticias sobre veículose espaciais e lançamento de mísseis

TABELA I
RESUMO DAS TAREFAS APRESENTADAS NAS DIVERSAS MUC

Existem actualmente em investigação e desenvolvimento cinco tarefas de IE (IE *tasks*), definidas pelas MUC, que são:

- Reconhecimento de nomes de entidades (NE do inglês *Name Entity*) que encontra e classifica nomes, locais, etc.
- Resolução de co-referências (CO) que identifica relações de identidade entre entidades nos textos.
- Identificação de Elementos de *Template* (*Template Elements* (TE)) que adiciona informação descritiva aos resultados da NE (utilizando CO).
- Construção de Relações entre *Templates* (TR) que encontra relações entre entidades TE.
- Produção de *Templates* de Cenários (ST de *Scenario Template*) que enquadra os resultados da TE e TR em cenários de eventos específicos.

IV. SISTEMAS DE IE

A IE é realizada através de sistemas automatizados que extraem uma determinada informação pertinente de um grande volume de textos em linguagem natural. Extraem informaçao pré-definida sobre entidades e relações entre essas entidades, colocando-a num modelo de base de dados estruturados.

Os sistemas de IE têm sido desenvolvidos para um leque de estilos de escrita que vai desde o texto estruturado com a informação organizada de uma forma tabular até ao texto livre. O elemento chave para este último tipo de escrita é a definição de um conjunto de regras de extracção que identificam a informação relevante a ser extraída.

Para texto estruturado, as regras especificam uma ordem fixa de informação relevante e os *labels* delimitam os caracteres que devem ser extraídos. Para texto livre (o estilo de texto utilizado nos relatórios médicos), um sistema de IE necessita de várias outras ferramentas para além das regras de extracção. Nestas ferramentas estão incluídas as de análise sintáctica, as de *tagging* semântico, os reconhecedores de objectos do domínio, tais como pessoas e nomes de companhias, e as de processamento de discurso que fazem inferências para além dos limites das frases. As regras de extracção para texto livre são tipicamente baseadas em modelos que envolvem relações sintácticas entre palavras ou com as classes semânticas das palavras.

Uma outra característica importante de um sistema de IE diz respeito ao facto de este extrair apenas factos isolados de um texto ou ter a capacidade de relacionar informação e extrair múltiplos campos relacionados. Existem algumas áreas de análise em que a extração multi-campo é essencial. Existem, no entanto, outros domínios em que a extracção de campos singulares é perfeitamente adequada. No caso em que existe sempre menos de um evento por texto os campos podem ser identificados separadamente, e posteriormente tratados como um único caso.

Anteriormente foram apresentadas as MUC e as suas tarefas, as quais foram responsáveis pelo interesse inicial na IE. Foi principalmente a partir destas tarefas que surgiram os sistemas de IE mais utilizados. Alguns destes sistemas, bem como as suas principais características, estão sistematizados na tabela II.

Muitos outros sistemas de IE são construídos a partir de cascatas de autómatos de estados finitos. O sistema FASTUS (*Finite State Automa-based Text Understanding System*) da SRI *International*[3] é um destes sistemas. O FASTUS, financiado pelo DARPA, tem actualmente uma

[1] A primeira MUC (MUC-1) foi realizada em 1987, a ultima conferência MUC-7 foi realizada em 1997

[2] http://www.itl.nist.gov/iaui/894.02/related_projects/muc

[3] http://www.ai.sri.com/appelt/fastus.html

Nome	Estilo de Texto	Multi-campo	Sintaxe
WI	estruturado	sim	não
SRV	semi-estruturado	não	não
RAPIER	semi-estruturado	não	não
AutoSlog	livre	não	sim
CRYSTAL	livre	sim	sim
CRYSTAL	semi-estruturado	sim	sim
LIEP	livre	só	sim
WHISK	estruturado	sim	não
WHISK	semi-estruturado	sim	não
WHISK	livre	sim	sim

TABELA II
COMPARAÇÃO ENTRE SISTEMAS DE IE QUE UTILIZAM APRENDIZAGEM POR REGRAS

avaliação para o reconhecimento de nomes de 92% de *recall* e 96% de *precision* (próximo da *performance* humana). Para a extracção de informação a avaliação é de 44% e 61%, respectivamente. Algumas das principais características deste sistemas são a utilização de uma linguagem declarativa para a especificação de regras gramaticais e a aprendizagem automática a partir de modelos.

Existem outros projectos de IE em desenvolvimento actualmente. Destes destacam-se o ALEMBIC da Workbrench (MITRE)[4], o Highlight da SRI Cambridge[5], o LaSIE/Gate da Universidade de Sheffield (apresentado em mais detalhe na secção VII), o NetOwl da SRA [1], o IDENTIFINDER [2] (BBN) (reconhecedor de nomes baseado em HMMs), o PROTEUS da Universidade de Nova York[6] e, por exemplo, o TextPro de Doug Appelt[7].

V. IE NA MEDICINA

A IE pode ser útil numa grande variedade de domínios. As várias MUCs focaram domínios como, por exemplo, o terrorismo latino americano e a microelectrónica. No entanto, a extracção de informação médica é um asunto que também pertence ao domínio actual da IE.

Um dos primeiros sistemas a utilizar informação proveniente de relatórios médicos como domínio de aplicação foi o BADGER que utiliza o CRYSTAL [3] como algoritmo de extracção. O principal objectivo deste sistema passa pela análise de relatórios médicos e pela identificação de referências a "diagnósticos" e a "sinais ou sintomas".

Actualmente o domínio bio-médico é bastante utilizado para o desenvolvimento de sistemas de IE. Por exemplo, o *Medstract*[8], criado devido ao aumento significativo de nova informação biológica, permite aceder rapidamente a nova informação pertinente e obter, desta forma, uma ideia do último conhecimento biológico. O objectivo da *Medstract* é aplicar os avanços recentes em linguística computacional e em análise de textos na extracção de informação de grandes bases de dados de informação bio-médica como é o caso da MedLine.

O *Unified Medical Language System (UMLS)* [4], desenvolvido na National Library of Medicine (NLM)[9], facilita o desenvolvimento de sistemas computacionais que se comportem como se "percebessem" o significado da linguagem da biomedicina e da saúde. Com este propósito a NLM produziu e disponibiliza as UMLS *knowledge sources* (bases de dados) e ferramentas associadas (programas) para serem utilizados na criação de sistemas de informação electrónicos que criam, processam, recuperam, integram e/ou agregam dados e informação biomédica e de saúde. Estes não estão optimizados para nenhuma situação em particular, mas podem ser aplicados em sistemas que executam um conjunto de funções envolvendo um ou mais tipos de informação, por exemplo relatórios de pacientes, literatura científica directrizes e dados de saúde pública.

Outros exemplos de sistemas de IE para o domínio biomédico são os que utilizam o sistema de IE GATE. Por exemplo, desde Outubro de 2004 que a Medwrite Inc.[10] utiliza o GATE no seu software para aplicações médicas. Mas este não é o único projecto de domínio médico a utilizar o GATE. O *Enzyme and Metabolic Path Information Extraction* (EMPathIE)[11] foi um projecto desenvolvido pelos departamentos de Estudos de Informação e de Ciências de Computadores da Universidade de Sheffield, cujo objectivo passava pela aplicação das tecnologias de IE às tarefas bioinformáticas. O EMPathIE reutilizou muitas das componentes existentes no GATE e produziu outros módulos baseados nos utilizados em projectos relacionados, de modo a extrair detalhes de reacções de enzimas de jornais biomédicos.

VI. ARQUITECTURA DE UM SISTEMA DE IE

De uma forma geral, os sistemas de extracção de informação são constituídos por quatro módulos principais: um *tokenizer*, algum tipo de processamento lexical e morfológico, análise sintáctica e módulos específicos do domínio em análise que identificam a informação a encontrar numa aplicação particular.

No entanto, dependendo dos requisitos de uma aplicação em particular, é desejável adicionar módulos a este esqueleto.

O esquema de um sistema de IE generalizado é ilustrado no gráfico de fluxo da figura 1.

Nesta figura, cada rectângulo grande representa uma componente. As caixas cinzentas representam as componentes que não são utilizadas em todos os sistemas de IE e são opcionais. As setas ilustram o fluxo do trabalho de IE, as setas interrompidas representam os caminhos opcionais. Os rectângulos mais pequenos representam recursos tais como léxicos, *corpora* de texto, bases de dados, listas de expressões comuns e listas de palavras.

Primeiro, o *corpus* de texto é itemizado (*tokenised*) em parágrafos, frases e palavras. Após a itemização procuram-se todas as palavras num dicionário lexical e se necessário

[4]http://www.mitre.org/tech/alembic-workbench/workbench-overview.html
[5]http://www.cam.sri.com/html/highlight_demo.html
[6]http://nlp.cs.nyu.edu/
[7]http://www.ai.sri.com/%7Eappelt/TextPro/
[8]http://www.medstract.org/
[9]http://www.nlm.nih.gov/
[10]http://medwrite.biz/
[11]http://www.dcs.shef.ac.uk/research/groups/nlp/funded/empathie.html

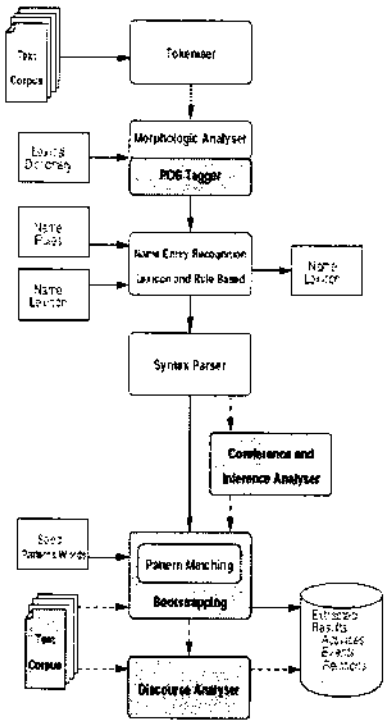


Fig. 1 - Esquema de um sistema de IE generalizado [5].

analisa-se a sua morfologia.

Alguns sistemas de IE aplicam na mesma fase os *Part-of-speech* (POS) *taggers*, com informação sintáctica adicional sobre as palavras, sob a forma de *tags*. A análise morfológica e o POS *tagging* estão amplamente relacionados e são frequentemente implementados como uma única componente.

A componente seguinte é denominada *Named Entity Recognition*. Os nomes das entidades consistem numa ou mais palavras e frequentemente representam informação a ser extraída. Existem vários métodos para o reconhecimento e extracção de nomes de entidades. Actualmente, o método mais aplicado é o reconhecimento de nomes de entidades com base em regras ou em léxico.

É possível adicionar mais informação sintáctica, comparável à informação proveniente da POS, através de *parsing* sintáctico. O *parsing* sintáctico não depende do reconhecimento de nomes de entidades e pode por isso ser aplicado também antes deste.

Os sistemas de IE recentes aplicam explicitamente análise de co-referência e inferências para produzirem melhores resultados. Alguns sistemas de IE pós-processam os resultados extraídos de modo a descobrir relacionamentos descritos no texto.

Encontrar modelos de extracção é a tarefa principal dos sistemas de IE. A informação é extraída utilizando estes modelos. Baseando-se na análise linguística efectuada sobre o texto, nas componentes descritas anteriormente, os modelos de extracção associam factos. Estes factos, ou peças de informação, são utilizados mais tarde para preencher os campos dos modelos dos resultados e são reunidos para formar tuplos de dados.

Os sistemas mais recentes usam vários algoritmos de *bootstrapping* para melhorar os resultados da associação de modelos, ou para fazer reconhecimento não supervisionado de nomes de entidades [5].

VII. GATE - *General Architecture for Text Engineering*

Esta secção descreve o sistema GATE, utilizado como software de base para o desenvolvimento do sistema de extracção de informação de relatórios médicos em Português.

O GATE é uma infraestrutura para o desenvolvimento e utilização de componentes de software que processam a linguagem humana. Em desenvolvimento na Universidade de Sheffield desde 1995, o GATE foi já utilizado numa grande variedade de pesquisas e projectos de investigação.

As componentes/recursos do GATE são tipos especializados de JavaBeans, mais especificamente, *Language Resources*, *Processing Resources* e *Visual Resources*.

Colectivamente, o conjunto de recursos integrados no GATE é denominado de CREOLE: *a Collection of REusable Objects for Language Engineering*.

Quando se utiliza o GATE para desenvolver funcionalidades de processamento de linguagem para uma aplicação, o investigador utiliza o ambiente de desenvolvimento para construir recursos dos três tipos. Isto pode envolver programação ou o desenvolvimento de Recursos Linguísticos tais como gramáticas que são utilizadas pelos "Processing Resources" existentes. É ainda possível uma mistura de ambas.

Quando um conjunto adequado de recursos tiverem sido desenvolvidos, podem ser embebidos na aplicação cliente alvo usando a estrutura do GATE.

A. *ANNIE* - A Nearly-New Information Extraction System

O GATE foi originalmente desenvolvido no contexto da investigação e desenvolvimento em IE. Vários sistemas de IE, em várias linguagens, de vários tamanhos e formas, foram criados utilizando o GATE com as componentes que foram distribuídas com este (ver [6] para a descrição de alguns destes projectos).

Uma família de *Processing Resources* para a análise linguística está incluída sob a forma de ANNIE, *A Nearly New Information Extraction System*.

Estas componentes utilizam técnicas de estados finitos para implementar várias tarefas desde a itemização até ao *tagging* semântico. Todas as componentes da ANNIE comunicam exclusivamente através dos documentos GATE e recursos de anotação.

O ANNIE é então uma família de recursos de processamento para análise linguística, tais como, *Tokeniser*, *Gazetteer*, *Sentence Splitter*, *Part-of-Speech Tagger*, *Semantic Tagger*, *Orthographic Coreference (OrthoMatcher)* e *Pronominal Coreference*.

B. *JAPE* - Java Annotation Patterns Engine

A linguagem JAPE permite o reconhecimento de expressões regulares sobre anotações em documentos.

Uma gramática JAPE consiste num conjunto de fases, em que cada uma é um conjunto de regras modelo/acção que

correm sequencialmente. Os modelos podem ser definidos pela descrição de um conjunto de caracteres específicos ou de anotações existentes (por exemplo, anotações criadas pelo *tokenizer, gazetteer, part-of-speech tagger*, ou pela análise do formato do documento). A definição de prioridades para as regras (se activada) previne a associação de multiplas anotações ao mesmo excerto de texto.

Até à data a JAPE foi utilizada com sucesso para o reconhecimento de nomes de entidades, *sentence splitting* e sumariação. Embora actualmente se utilizem regras produzidas manualmente, deve ser possível para uma aplicação aprender regras automaticamente.

As fases que constituem a gramática JAPE correm sequencialmente e constituem uma cascata de transdutores de estado finito sobre anotações. O lado esquerdo (LHS) das regras consiste em modelos de anotações que podem conter operadores de expressões regulares (ex.: *, ?, +). O lado direito (RHS) é constituído por expressões de manipulação de anotações. As anotações associadas pelo LHS de uma regra podem ser referidas no RHS através de etiquetas que são adicionadas aos elementos do modelo.

O exemplo seguinte apresenta uma regra gramatical para a extracção de um endereço de e-mail (assumindo uma definição apropriada de (EMAIL)) no caso de este ocorrer entre os sinais de menor e maior, respectivamente.

```
Rule: Emailaddress1
({Token.string == ''<''})
(
 (EMAIL)
)
:email
({Token.string == ''>''})
-->
 :email.Address= {kind = "email",
                  rule = "Emailaddress1"}
```

VIII. ADAPTAÇÃO AO PORTUGUÊS

Para o desenvolvimento do sistema de IE de relatórios médicos foi utilizado o GATE. Esta escolha é justificada pela sua simplicidade e capacidade de integração. No entanto, alguns dos recursos existentes neste sistema não são úteis para a extracção de informação em Português.

São as alterações já efectuadas a este sistema, com o objectivo de o adaptar à IE em Português que preenchem esta secção.

As primeiras experiências efectuadas com o GATE mostram que ferramentas como o *Tokeniser* e o *Sentence Splitter* podem ser utilizadas para a IE em Português. Assim, a tarefa de extracção de informação dos relatórios médicos começa com a utilização de um *Tokeniser* e de um *Sentence Splitter* cujos parâmetros/regras são as definidas no ANNIE.

Por outro lado, todos os resultados originados, quer pelo POS *tagger*, quer pelas listas de *Gazetteer* e mesmo pelas gramáticas semânticas, não correspondem, de uma forma geral, a resultados correctos. Este facto não é surpreendente uma vez que tais ferramentas baseam-se em conceitos linguísticos específicos da língua inglesa que não podem, por isso, ser aplicados ao Português.

A. *VMP Tagger*

Optou-se pela substituição do POS *Tagger* utilizado no ANNIE por um desenvolvido por Valentina Munóz em 2005 e disponível em *http://sourceforge.net/projects/vmptagger/*. Este POS *tagger* desenvolvido em Java e baseado no Brill Tagger [7], foi desenvolvido para a utilização integrada no GATE e para a categorização morfo-sintáctica em qualquer língua. Para tal apenas é necessária a especificação de quatro listas correspondentes a um léxico, um ficheiro de regras lexicais e outro de regras contextuais e um bigrama, que podem ser obtidas através do treino de pequenos *corpus* etiquetados na língua em análise.

As listas utilizadas neste trabalho são originárias de um etiquetador morfo-sintáctico desenvolvido na Universidade do Minho, que é descrito de seguida.

A.1 *Conjunto de* tags

Sendo a Língua Portuguesa de origem latina, tem uma relativa complexidade morfológica. A definição de um conjunto de *tags* para o Português é uma tarefa complicada e fulcral e corresponde também a um compromisso entre a precisão na descrição e a capacidade de aprendizagem de regras. Decidiu-se assim utilizar o conjunto de *tags* definido por Reis et al. [8]. Este conjunto segue alguns princípios funcionais tais como a precisão, a definição de uma estrutura hierárquica para cada etiqueta e baseia-se em modelos já utilizados em experiências com outras línguas.

A nomenclatura é de alguma forma hierárquica, o que implica que cada campo da *tag* tem um significado específico e estes campos são o mais reduzidos possível.

As *tags* são definidas com a seguinte estrutura:

Etiqueta: Categoria Sub-Categorias Género Pessoa Número

Categoria: QUE || SE || D || P || N || J || V || ADV || C || I || & || ?

referentes respectivamente a **que**, **se**, **D**eterminantes, **P**ronomes ou **P**reposições, **N**omes, ad**J**ectivos, **V**erbos, **ADV**érbios, **C**onjunções, **I**nterjeições, **C**ontracções e palavras desconhecidas.

As sub-categorias dependem, naturalmente, de cada categoria sendo os restantes elementos flexíveis dentro dos espaços conhecidos.

A categoria é o único campo obrigatoriamente preenchido. No caso de preposições ou interjeições será mesmo o único a ter valor.

B. *Gazetteer*

A alteração das listas do *Gazetteer* foi efectuada de modo a reflectir conceitos da língua portuguesa. Foram também adicionadas listas específicas do domínio, tais como listas correspondentes a nomes de doenças, a nomes de exames da área da Electroencefalografia, e a características dos resultados dos exames, entre outras.

C. *Gramáticas JAPE*

Para a extracção da informação do domínio pretendida foram desenvolvidas várias gramáticas baseadas na linguagem JAPE. Como exemplo referem-se as gramáticas para a

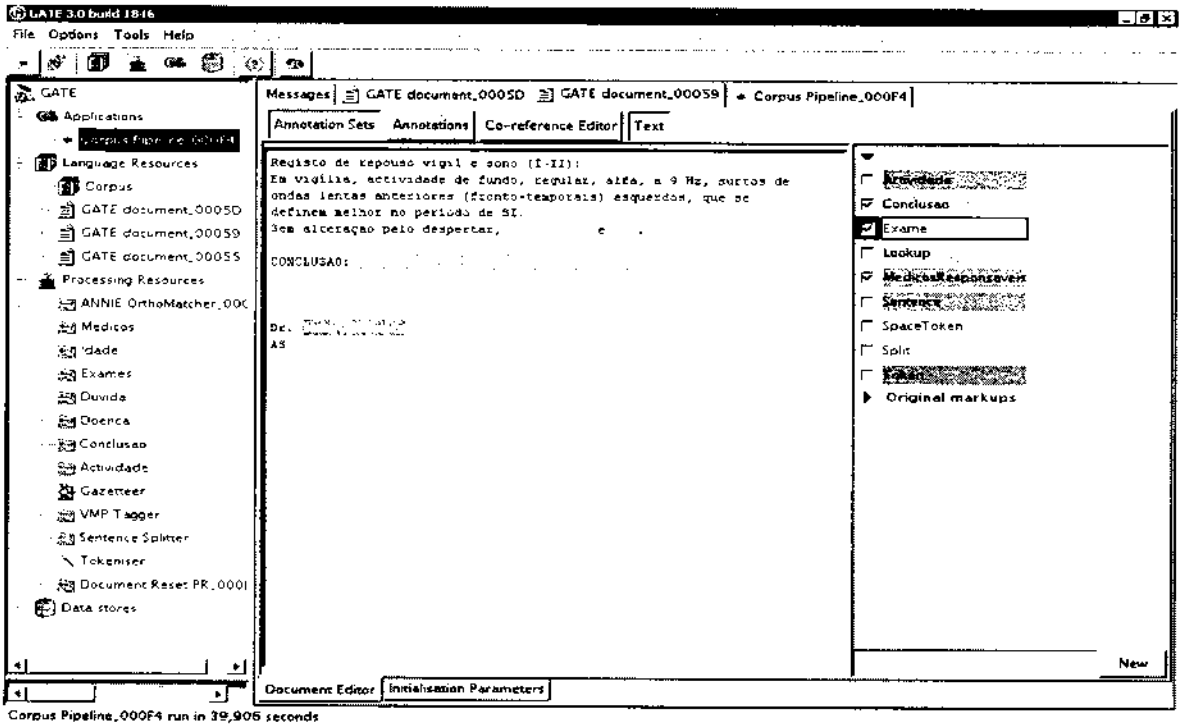


Fig. 2 - Elementos das entidades *MédicosResponsáveis*, *Conclusão* e *Exame*

extracção de nomes de doenças, características das actividades descritas nos relatórios, extracção de informação existente na conclusão destes, entre outras. Uma regra desenvolvida para a extracção do nome dos médicos responsáveis pelo exame, originando, deste modo, a entidade *Medicos-Responsaveis*, é apresentada de seguida.

```
/*
* When this grammar rule is invoked the
* rule Medicos is then invoked and this
* recognizes the lookup with majorType
* title (e.g. Dr.) followed by an space
* token. Following the rule Dr. Joao
* Lopes it will be annotated as a
* MedicosResponsaveis entity of type
* Name.
*/

Phase: medicos
Options: control = brill
Rule: Medicos
//e.g. Dr. Joao
({Lookup.majorType==title}
{SpaceToken}
)
(
{Token.orth == upperInitial}
({Token.string == "."})?
{SpaceToken}
{Token.orth == upperInitial}
)
:medicos -->
:medicos.MedicosResponsaveis
= {kind="name", rule="Medicos"}
```

IX. EXEMPLO

Um exemplo do resultado originado pela gramática anterior, em conjunto com todas as outras ferramentas apresentadas, pode ser analisado na figura 2.

Nesta figura é possível visualizar o ambiente de desenvolvimento do GATE bem como os recursos utilizados, em particular os *Language Resources* e os *Processing Resources* (lado esquerdo da figura). Na zona central encontra-se um exemplo de relatório utilizado e algumas das anotações efectuadas sobre este, concretamente as anotações que delimitam algumas das entidades consideradas importantes, como as entidades *MédicosResponsáveis*, *Conclusão* e *Exame*.

X. CONCLUSÃO

O sistema apresentado neste artigo resulta, quer de adaptações efectuadas a ferramentas de extracção de informação existentes no sistema GATE, quer da implementação/criação de novas ferramentas direccionadas para a IE em Português e de relatórios médicos da área da Neurofisiologia. São exemplo destas últimas algumas das gramáticas desenvolvidas.

Actualmente o sistema determina correctamente informação relativa ao conjunto de entidades consideradas relevantes no domínio. A inclusão de módulos que correlacionem as entidades identificadas, bem como o desenvolvimento de gramáticas para a resolução de anáforas, são algumas das tarefas a realizar brevemente, que poderão melhorar o desempenho do sistema.

BIBLIOGRAFIA

[1] Chinatsu Aone, Lauren Halverson, Tom Hampton, e Mila Ramos-Santacruz, "Sra: description of the ie2 system used for muc-7", em *Seventh Message Understanding Conference (MUC-7)*, San Francisco, California, 1998, Morgan Kaufmann Publishers.

[2] Scott Miller, Michael Crystal, Heidi Fox, Lance Ramshaw, Richard Schwartz, Rebecca Stone, Ralph Weischedel, e Annotation group, "Algorithms that learn to extract information; bbn: Description of the sift system as used for muc-7", em *Seventh Message Understanding Conference (MUC-7)*, San Francisco, California, 1998, Morgan Kaufmann Publishers.

[3] Stephen Soderland, David Fisher, Jonathan Aseltine, e Wendy Lehnert, "CRYSTAL: Inducing a conceptual dictionary", em *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, Chris Mellish, Ed., San Francisco, 1995, pp. 1314–1319, Morgan Kaufmann.
URL: *citeseer.csail.mit.edu/soderland95crystal.html*

[4] D. Lindberg, B. Humphreys, e A. McCray, "Unified medical language systems", *Methods of Information in Medicine*, vol. 32, no. 4, pp. 281–291, 1993.

[5] Philipp Johannes Masche, "Multilingual information extraction", Master's thesis, Dept. of Computer Science, Faculty of Science, University of Helsinki, 2004.

[6] Diana Maynard, Hamish Cunningham, Kalina Bontcheva, Roberta Catizone, George Demetriou, Robert Gaizauskas, Oana Hamza, Mark Hepple, Patrick Herring, Brian Mitchell, Michael Oakes, Wim Peters, Andrea Setzer, Mark Stevenson, Valentin Tablan, Christian Ursu, e Yorick Wilks, "A survey of uses of gate", Relatório Técnico CS–00–06, Department of Computer Science, University of Sheffield, 2000.

[7] Eric Brill, "A simple rule-based part-of-speech tagger", em *Proceedings of ANLP-92, 3rd Conference on APlied Natural Language Processing*, Trento, Itália, 1992, pp. 152–155.

[8] Ricardo Reis e José João Dias de Almeida, "Etiquetador morfo-sintáctico para o português", em *Actas do XIII Encontro da Associação Portuguesa de Linguística*, Lisboa, Portugal, 1997, vol. 2, pp. 209–222, Associação Portuguesa de Linguística.

Language Models in Automatic Speech Recognition

Ciro Martins, António Teixeira, João Neto[†]

Resumo - **O presente artigo descreve o trabalho desenvolvido com o objectivo de melhorar o desempenho da componente modelo de linguagem de um sistema de reconhecimento de fala contínua para a língua Portuguesa. Como modelo de base, utilizou-se um sistema de reconhecimento de fala de grandes vocabulários desenvolvido para uma tarefa de reconhecimento de notícias (Broadcast News). Foram analisadas duas metodologias diferentes com o objectivo de aumentar a eficácia do sistema: a utilização de maiores quantidades de dados para melhor estimação dos parâmetros associados ao modelo de linguagem, e a utilização de diferentes técnicas de "pruning" e "discounting" dos referidos parâmetros. Os resultados mostram que com a utilização de maiores quantidades de dados se obtiveram ligeiras melhorias a nível da taxa de eficácia de reconhecimento (cerca de 5%). Aplicando uma técnica de "pruning" baseada no conceito de entropia, obteve-se uma redução significativa da dimensão do modelo de linguagem (reduções de 30% ou mais), com um ligeiro incremento dos valores da perplexidade e taxa de erro ao nível da palavra.**

Abstract - **In this paper we describe the work done with the updating and improvement of the language model component of a continuous speech recognition system for the Portuguese language. As a baseline system we used a large vocabulary speech recognition system for the Portuguese language, developed for a Broadcast News (BN) recognition task. Two sources of performance improvement have been studied: the inclusion of more training data to better estimate the language model parameters, and the use of different discounting and pruning techniques. The results show that using more training data helped to achieve a small relative improvement in recognition accuracy (about 5%). Applying an entropy based pruning technique one can get up to more than 30% size reduction with a slightly increase on perplexity and WER.**

I. INTRODUCTION

Statistical language modeling has many applications in a large variety of areas, including speech recognition, optical character recognition, machine translation, spelling correction, etc. Despite all the research done on the last two decades, N-gram language models still dominate as the technology of choice for state-of-the-art speech recognizers.

Typically, N-gram models for large vocabulary speech recognizers are trained on hundred of millions or billions of word strings. In constructing such kind of models, we usually face two problems. First, the large amount of training data can lead to models too large for real applications. On other hand, to train a specific domain model, we must deal with the data sparseness problem, because large amount of domain specific data are not available.

To overcome this kind of problems, many different approaches have been suggested. Smoothing techniques are usually used to better estimate probabilities when there is insufficient data to estimate probabilities accurately [1]. In case where small amount of in-domain data is available, the use of mixture of language models by means of linear interpolation proved to increase the quality of language models [2]. On the other side, some form of model size reduction is critical for practical applications, especially when the model is trained with large amount of data. Many different pruning techniques have been proposed which leads to significant model size reduction without decreasing their performance [3] [4].

In the reminder of this paper we describe the work done with the updating and improvement of the language model component of a continuous speech recognition system for the Portuguese language. As a baseline system we used the work presented in [5] that we briefly describe in section 2. In section 3 we describe the datasets we have used to train and evaluate the new models we obtained applying some of the techniques referenced before. In section 4 we summarize some results in terms of perplexity and word error rate (WER), drawing in section 5 and 6 some conclusions and future work to be done.

II. BASELINE SYSTEM

As the starting point for the work presented in this paper, we used the same datasets and system reported in [5]. This is a large vocabulary speech recognition system for the Portuguese language, used for a Broadcast News (BN) recognition task.

[†] L2F – Spoken Language Systems Lab; INESC-ID/IST, Lisbon.

A. Acoustic Component

The baseline recognizer AUDIMUS [6], used in this task, is a hybrid HMM/MLP system. It uses three MLPs, each of them associated with a different feature extraction process, where the MLPs are used to estimate the context-independent posterior phone probabilities given the acoustic data at each frame. The phone probabilities generated at the output of the MLPs classifiers are combined using an appropriate algorithm [6]. All MLPs use the same phone set constituted by 38 phones for the Portuguese language plus the silence and breath noises. The training and development of this system was based on the European Portuguese ALERT BN database (ALERT BN) [5]. The train of the recognition system was done using 45 hours of audio data. For system evaluation there exist two different sets: a development set comprising of approximately 6 hours and half, and an evaluation set comprising of 4 hours and half of audio data.

B. Language Model Component

The language model component of this system has been created using two different sources (table 1): a text corpus collected daily from the online Portuguese newspapers Web editions, and the training set transcriptions of ALERT BN database.

Source	# Sentences	# Words
newspapers set	24.0M	434.4M
ALERT BN training set	9K	142K

Table 1: Size of the corpuses used in language model training

At this time the newspapers set included texts collected from different Portuguese newspapers ("A Bola", "Diário de Notícias", "Diário Económico", "Expresso", "Expresso Diário", "Jornal de Notícias", "O Jogo", "O Independente", "O Público"), since 1991 until the end of 2001. The ALERT BN transcriptions used to train the language model component include only part of the ALERT BN training transcriptions available at that time. The complete and final set is the one described in the next section.

From this two sources and using the CMU Cambridge Toolkit [7], two different language models have been generated. From the newspapers set a backoff 4-grams LM has been generated using the absolute discounting method and applying cutoff values of 2, 3 and 4 respectively for 2-grams, 3-grams and 4-grams. From the ALERT BN training set a backoff 3-grams LM has been generated using the absolute discounting method without applying any kind of cutoffs. Finally, the two models were combined by means of linear interpolation, generating a mixed model referred here as MIX_BASELINE. The optimal interpolation weights obtained were 0.829 for the newspapers set component and 0.171 for the ALERT BN training set component.

C. Vocabulary and Lexicon

Currently the vocabulary size is limited to 57,564 words (referred here as 57k). This vocabulary was first created using 56k different words selected from the newspapers set according to their weighted class frequencies of occurrence. Different weights were used for each class of words. All new different words present in the training data transcriptions of ALERT BN database were added to the vocabulary giving a final vocabulary of 57,564 words.

The pronunciation lexicon was built from the vocabulary, giving a total of 65,585 different pronunciations.

D. Dynamic Decoder

The decoder used under this baseline system is based on a weighted finite-state transducer (WFST) approach to large vocabulary speech recognition [8]. In this approach, the decoder search space is a large WFST that maps observation distributions to words. This WFST consists of the composition of various transducers representing components such as: the acoustic model topology H; context dependency C; the lexicon L and the language model G. The search space is thus $H \circ C \circ L \circ G$, which is built "on-the-fly" in opposition to traditional approaches that compile it outside of the decoder, using it statically during the decoding process.

III. NEW DATASETS FOR LANGUAGE MODELING

To update and improve the language model component of the baseline system described before, we have collected more texts from the newspapers online editions until the end of 2003, and used the final ALERT BN training set that is now available. In table 2 we summarize the size of these new sets that have been used to generate and test the new language models we have developed. For the new ALERT BN training set which has 26,715 different words in a total of 531,757 words, the number of Out-Of-Vocabulary words (OOVs) using the 64k word vocabulary is 6138, representing an OOV word rate of 1,15%.

Source	# Sentences	# Words
newspapers set	38.8M	604.2M
ALERT BN training set	34K	531.7K

Table 2: Size of new training sets used in language model training

To evaluate the language models performance we used the ALERT BN evaluation set. To estimate some parameters like the ones necessary for the linear interpolation process we used the ALERT BN development set as a held-out corpus. In table 3 we describe these two corpuses.

Source	# Sentences	# Words	Duration (audio)
development set	4,194	66,495	6h 24m
evaluation set	3,125	47,473	4h 30m

Table 3: Size of ALERT BN development and evaluation sets

For the development set which has 8,538 different words in a total of 66,495 words, the number of OOVs words (OOVs) using the 57k word vocabulary is 879, representing an OOV word rate of 1.32%. The Evaluation set has 7,000 different words in a total of 47,473 words, having 675 OOVs, which represents an OOV word rate of 1.42%.

IV. EXPERIMENTAL RESULTS

The most common metric for evaluating a language model is perplexity. It is often used as a language model quality measure as it tests its capability to predict an unseen text, i.e., a text not used in model training. Formally, the word perplexity PP of a model relative to a text with n words is defined as:

$$PP = 2^{-\left(\frac{1}{n}\right)\log P(w_1 \ldots w_n)} \quad (1)$$

However, perplexity metric does not take into account the acoustic similarity between words. This means that lower perplexity values may not result in lower word error rate (WER) during the recognition process. For that reason, it is usual to use WER as another metric to evaluate the language model performance over all the recognition system.

For the experimental results we present in this paper we used both metrics to consistently evaluate and compare the relative language models performance. The reported results were conducted in the ALERT BN Evaluation set, using the ALERT BN Development set as a held-out set to estimate and optimize some parameters like the ones used by the interpolation process.

To generate the language models used in this work and evaluate their performance in terms of perplexity values, we used the SRI Language Modeling Toolkit (version 1.4) [9].

All the experiments were done using the same 57,564 word closed-vocabulary. End-of-sentence symbols were included in perplexity computations, but out-of-vocabulary words were not. Related with recognition results, we used all the evaluation set, which means the results take into account the effect of OOVs during the recognition process, i.e., since we are using a closed-vocabulary all the OOVs are misrecognized by the system.

A. Perplexity Results

First of all, we started by computing the perplexity value for the baseline language model (MIX_BASELINE) using the SRI Toolkit. For this model we obtained a perplexity value of 117.5. This was a reference value, being used to make performance comparison to the new language models. In [5] one can realize a different perplexity value of 139.5 for the same baseline language model, which is due to the fact that we are now using SRILM Toolkit instead of CMU Toolkit. These toolkits treat sentences clues in a different way when measuring text perplexities.

To evaluate the effect of using more data we trained the language models (the newspapers model and the ALERT BN model) based on the new training sets. This train was done using the same conditions as the baseline ones, i.e., we used the same discounting method (absolute discounting), the same model order (4-grams for the newspapers model and 3-grams for the ALERT BN model) and the same cutoff values.

For the newspapers language model we generated two different versions: one using the newspapers data available until the end of 2001 (referred as NP_2001; line 1 of table 1) and another one using all the newspapers data available until the end of 2003 (referred as NP_2003; line 1 of table 2). For the ALERT BN language model, and since we didn't have the partial training set used to generate the baseline ALERT BN model, we only generated one version using the final ALERT BN training set (referred as ALERT_BN_ALL; line 2 of table 2). Finally we generated two mixed models: one using NP_2001 and ALERT_BN_ALL (referred as MIX_2001) and another one using NP_2003 and ALERT_BN_ALL (referred as MIX_2003).

From table 4 we can realize a small decrease (less than 2%) on the perplexity value when we used more training data. Comparing baseline LM perplexity to the new models perplexity we conclude that the biggest improvement (almost 4.5%) is due to the use of more training data related with the domain. In fact, the final ALERT BN training set is 4 times bigger that the one used to generate the baseline model (MIX_BASELINE). The column "Param" gives the number of stored N-grams (only the last order).

Models	Interpol. weights		PP	Param
	α_{NP}	α_{BN}		
MIX_BASELINE	0.829	0.171	117.5	6,731,820
NP_2001	-	-	122.5	6,741,258
NP_2003	-	-	121.0	9,904,128
ALERT_BN_ALL	-	-	335.2	364,004
MIX_2001	0.816	0.184	114.1	6,741,258
MIX_2003	0.814	0.186	112.3	9,904,128

Table 4: Comparison of Baseline LM perplexity vs. New LMs perplexities

Taking into account the experiments described in [10] we decided to investigate the results we would achieve by applying a modified interpolated form of Kneser-Ney discounting method [1]. Kneser-Ney smoothing uses a modified backoff distribution based on the number of

contexts each word occurs in, rather than the number of occurrences of the word. In [10] it is showed that a modified interpolated form of Kneser-Ney smoothing outperformed other smoothing techniques.

Models	PP	
	Absolute disc.	Kneser-Ney disc.
ALERT_BN_ALL	335.2	299.1
NP_2003	121.0	122.7

Table 5: Kneser-Ney discounting method vs. absolute discounting method

Table 5 shows that in case where a small quantity of data is available we can get better results by applying the interpolated Kneser-Ney discounting method (for ALERT BN model we obtained a perplexity reduction of about 11%). However, for large data training sets, as the newspapers one, we didn't get advantage in applying Kneser-Ney method. Mixing newspapers model obtained with absolute discount and ALERT BN model obtained with Kneser-Ney discount we get an interpolated model (referred here as MIX_2003_BEST) with a perplexity value of 111.4, our best result. The optimal interpolation weights obtained were 0.796 for the newspapers set component and 0.204 for the ALERT BN training set component.

Finally, we investigated the effects of pruning language models using an entropy-based pruning technique [3], i.e., pruning all n-grams that would increase the relative perplexity by less than a given threshold. Simultaneously, we pruned all the n-grams having probabilities lower than the corresponding backed-off estimates. This last pruning is useful to generate models that can be correctly converted into probabilistic finite-state grammars.

For this experiment we used the best mixed language model (MIX_2003_BEST). Table 6 shows model size and perplexity results obtained with various pruning thresholds. As shown, pruning is highly effective. For a threshold of 1e-09 we obtain a model that is about 30% the size of the original model without significant degradation of perplexity. On the following point we present the results in terms of WER.

Threshold	PP	Param	Size (.gz)
no pruning	111.4	9,904,128	303.1 Mb
1e-09	112.9	4,887,956	210.6 Mb
1e-08	119.7	1,511,364	102.1 Mb
1e-07	150.4	102,878	18.4 Mb

Table 6: Perplexity as a function of pruning threshold and language model size

B. Speech Recognition Results

Speech recognition experiments were conducted in a Pentium IV 2.8GHz computer with 2Gb RAM running Linux. The experiments were done under the same conditions, only varying the language model used. For these experiments we used the absolute discount version of NP_2003 model and the Kneser-Ney discount version of ALERT_BN_ALL. Table 7 summarizes the word error rate (WER) obtained for the different language models using the baseline system described in section 2. In this work we used the parametric conditions defined in line 6 of table 3 presented in [5]. However, we can not directly compare the baseline WER (26.5%) obtained in [5] since it was based on the development test set. For that reason, we evaluated the baseline language model over the evaluation test set, getting a WER of 28.2%, as expressed in line 1 of table 7.

Models	%WER	xRT
MIX_BASELINE	28.2	2.4
NP_2003	27.8	2.3
ALERT_BN_ALL	37.2	0.8
MIX_2003_BEST	26.9	2.4

Table 7: Speech recognition results as a function of language model

From line 4 of table 7 we can realize a small WER relative improvement of about 5% when comparing to the mixed baseline model. Finally, we evaluated the pruning effect on the WER. For this propose we used again the best mixed language model (MIX_2003_BEST). The results are summarized in table 8.

Pruning Threshold	%WER	xRT
no pruning	26.9	2.4
1e-09	26.7	2.0
1e-08	27.3	1.6
1e-07	29.4	1.1

Table 8: Pruning effect in speech recognition results

The results show that language models can be reduced up to 30% of its original size without significantly affecting the recognition accuracy. In this case we were able to get real-time decoding performance with only a small increase in word error rate, generating a language model 94% times smaller than the unpruned one.

V. CONCLUSIONS

From the results obtained in our experiments we concluded that in case where small data corpuses are available one get better results using a modified interpolated form of Kneser-Ney discounting method instead of absolute discounting.

In this work we increased the general domain data training set from 434.4 million words to 604.2 million words and we were able to obtain only a relative recognition error

decrease of about 5%. This suggests us that one should try to investigate other kind of approaches to improve the language model component, especially the ones related to domain adaptation of language models.

The applied entropy based pruning algorithm is highly effective. For a pruning threshold equal to 1e-09, we obtained a model that is 30% smaller than the original one without degradation in recognition performance (a slightly decrease in WER and a speed-up of about 17% in decoding time).

V. FUTURE WORK

As future work we will investigate the use of different clustering techniques applied to the Portuguese language, using class-dependent language modeling. Using these techniques we hope to get improvements not only in WER but mainly in language model size reduction, which will permit us to increase the system vocabulary size without compromise its practical level.

REFERENCES

[1] Chen, S. and Goodman, J., "An Empirical Study of Smoothing Techniques for Language Modeling", Computer Science Group - Harvard University, Cambridge, Massachusetts TR-10-98, 1998.

[2] Rosenfeld, R., "Adaptive Statistical Language Modeling: A Maximum Entropy Approach", PhD Thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, 1994.

[3] Stolcke, A., "Entropy-based Pruning of Backoff Language Models", in Proc. DARPA News Transcription and Understanding Workshop, Lansdowne, VA, 1998.

[4] Goodman, J. and Gao, J., "Language Model Size Reduction by Pruning and Clustering", in Proc. ICSLP 2000, Beijing, China, 2000.

[5] H. Meinedo, D. Caseiro, J. Neto, and I. Trancoso, "AUDIMUS.MEDIA: A Broadcast News Speech Recognition System for the European Portuguese Language", presented at PROPOR 2003 - VI Encontro para o Processamento Computacional do Português Escrito e Falado, Faro, Portugal, 2003.

[6] Meinedo, H. and Neto, J., "Combination of Acoustic Models in Continuous Speech Recognition Hybrid Systems", in Proc. ICSLP 2000, Beijing, China, 2000.

[7] Clarkson, P. and Rosenfeld, R., "Statistical Language Modeling using the CMU-Cambridge Toolkit", in Proc. EUROSPEECH 97, Rhodes, Greece, 1997.

[8] Caseiro, D., "Finite-State Methods in Automatic Speech Recognition". Lisbon, Portugal: Instituto Superior Técnico, Universidade Técnica de Lisboa, 2003.

[9] Stolcke, A., "SRILM - An Extensible Language Modeling Toolkit", in Proc. International Conference on Spoken Language Processing, Denver, USA, 2002.

[10] Goodman, J., "A Bit of Progress in Language Modeling - Extended Version", Machine Learning and Applied Statistics Group - Microsoft Research, Redmond, WA MSR-TR-2001-72, 2001.

Análise Digital de Radiografias Dentárias

Luís Coelho*, Augusto Silva

Abstract – **Digital image analysis is now considered an important complementary tool for medical diagnosis in a wide variety of instances. Several pathologies can be detected and classified with computer aided support. Dentistry uses the dental x-rays as an aid in their diagnoses. In this paper we describe several digital image analysis techniques focused on the identification of teeth contours. Contour information is then post-processed in order to provide quantitative parameterization of the degree of periodontitis In our work we implemented three techniques of analysis. The first method uses the gradient information, the second method uses probabilistic models of the intensity distribution, and finally, the third method uses the Deformable models (GVF Snakes). Special attention is given to the initialization modes used in each technique. Finally we show the experimental results of each method regarding the extraction of the crown contours and the detection of the gum level.**

Resumo – **A análise digital de imagens constitui um meio complementar que facilita o diagnóstico de diversas patologias, em diversas áreas. A Odontologia usa as radiografias dentárias para auxiliar nos seus diagnósticos. Neste artigo estão descritas algumas técnicas de análise digital de radiografias dentárias, usadas para a extracção dos contornos de dentes, com o objectivo de apoiar o diagnóstico de determinadas patologias. Foram implementadas três técnicas de análise, a primeira usa a informação do gradiente da imagem, a segunda usa modelos probabilísticos da distribuição de intensidades e a terceira usa os modelos deformáveis (GVF Snakes). Após a apresentação sucinta das técnicas implementadas, descrevem-se os modos de inicialização usados em cada uma das técnicas. Segue-se a descrição dos algoritmos de validação dos pontos do contorno da coroa e da detecção do nível da gengiva. Finalmente são apresentados os resultados obtidos com a aplicação destas técnicas em imagens reais.**

Keywords – **Digital Image Analysis, Dentistry, Priodontitis**

Palavras chave – **Análise Digital de Imagem, Radiografias Dentais, Periodontite**

I. INTRODUÇÃO

As técnicas imagiológicas assumem-se cada vez mais relevantes no universo dos meios complementares de diagnóstico médico. No sentido de maximizar a acuidade do diagnóstico, procura-se correlacionar a informação imagiológica com outros dados clínicos. A odontologia recorre, também, à radiologia projectiva para auxiliar os seus diagnósticos. Um dos principais objectivos do uso do computador é eliminar os erros associados à actividade humana e recorrer à velocidade e reprodutibilidade algorítmica dos computadores, para garantir, de forma eficaz, o máximo de rigor ao diagnóstico. Com a digitalização das radiografias, e com o auxílio de um computador, pode-se visualizar e alterar a imagem a qualquer momento, melhorando a perceptibilidade; visualizar áreas ampliadas; comparar as alterações efectuadas com a imagem original e armazená-las em discos rígidos para que possam ser consultadas posteriormente. Além disso, as imagens poderão ser reproduzidas e transmitidas para outros locais distantes, de maneira rápida e segura. A troca de informações clínicas entre várias partes, promove o debate sobre o estado clínico do paciente, podendo aumentar o grau de certeza do diagnóstico.

Recentemente surgiram algumas publicações, que referem a aplicação de algumas técnicas de análise digital na identificação humana através de radiografias dentárias. Jain et al. [1], [2], refere a aplicação de técnicas de análise e processamento digital de imagem e de reconhecimento de padrões, com o intuito de reconhecer a identidade de cadáveres. É feita uma análise a radiografias dentárias do cadáver, e partindo dessa análise, procede-se à pesquisa numa base de dados de radiografias dentárias, de modo a identificar a identidade da pessoa em causa. Estas técnicas de extracção dos contornos do dente são divididas em duas etapas. A extracção do contorno da coroa, e a extracção do contorno da raiz. Muito recentemente, Chen e Jain [3], propuseram um método baseado em modelos de contorno activo também conhecidos por Snakes. O método proposto é denominado por Directional Snake [4], que tem como base a direcção da variação do gradiente da imagem. Neste trabalho foi implementado um método conhecido por Gradient Vector Flow - Snakes (GVF - Snakes), proposto originalmente por Xu e Prince [5]-[7]. Apesar de serem várias as aplicações conhecidas para estes métodos, ainda não foram encontradas referências em relação ao uso destes métodos na identificação e avaliação de patologias do foro estomatológico. Este artigo começa por descrever as características de cada um dos métodos implementados; Segue-se a descrição dos modos de inicialização permitidos para cada um dos métodos descritos, aos quais se segue a descrição do algoritmo de validação dos contornos da coroa e da detecção dos níveis da gengiva. Finalmente é feita a exposição dos resultados obtidos e são apresentadas algumas conclusões, assim como algumas sugestões para trabalhos futuros.

II. EXTRACÇÃO DO CONTORNOS DO DENTE

No nosso trabalho, implementámos três métodos para a detecção dos contornos do dente. Dois deles procedem à detecção individual de cada uma das partes principais do

*Financiado pela Unidade de Investigação 127/94 IEETA da Universidade de Aveiro.

dente (a coroa e a raiz). O primeiro usa a informação do gradiente da imagem e o segundo usa modelos probabilísticos da distribuição dos níveis de intensidade. O terceiro método usa os modelos deformáveis (Snakes) e trata o dente como um todo, detectando a totalidade do seu contorno, que posteriormente é separado nas suas partes constituintes. Segue-se a descrição dos algoritmos usados, por cada um dos métodos.

A. *Método do Gradiente*

Este método, sugerido por Jain et al. [2], pode ser dividido em duas etapas, a etapa de detecção da coroa e a etapa de detecção da raíz. Por norma, as radiografias dentárias não apresentam uma boa definição e os dentes nela representados apresentam várias alterações na sua morfologia, tornando muito difícil o desenvolvimento de um sistema de segmentação, totalmente automático. Para facilitar a tarefa de detecção dos contornos do dente, é necessário que o utilizador marque aproximadamente o centro da coroa, C, um rectângulo R em torno do dente que pretende segmentar e um ponto que defina o raio de varrimento para determinar os pontos do contorno da coroa. A partir desta selecção, o sistema detecta automaticamente os contornos o dente assinalado. Na fig. 1(a) está ilustrado um exemplo dos pontos a assinalar, na inicialização do sistema. O algoritmo de detecção da coroa inicia-se com o cálculo do gradiente da imagem,$|\nabla I|$ da imagem I. (ver fig. 1(b)). Na zona de contacto entre o dente em análise e os dentes adjacentes existe interferência nos contornos. Existe uma técnica que elimina essa interferência e que consiste em aplicar uma máscara B à imagem gradiente. O resultado é a imagem auxiliar M (fig. 1(c)), determinada por,

$$M(x,y) = B(x,y)|\nabla I(x,y)| \tag{1}$$

onde

$$B(x,y) = \begin{cases} 0 & \text{se } \langle \nabla I(x,y), E(x,y) \rangle < 0 \\ 1 & \text{restantes casos} \end{cases} \tag{2}$$

sendo $\nabla I(x,y)$ o vector gradiente no ponto (x,y), $E(x,y)$ o vector definido pelo centro da coroa C e pelo ponto (x,y), onde $\langle .,. \rangle$ representa o produto interno.

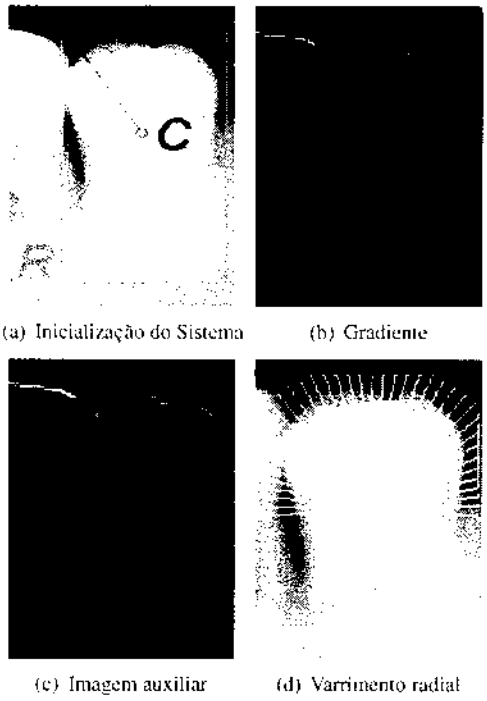
Após este processamento, inicia-se o processo de detecção do contorno. Este processo obedece aos seguintes passos:

- Partindo do centro da coroa, C, efectua-se um varrimento radial de modo a determinar os pontos candidatos a contorno da coroa. Ver fig. 1(d);
- Ordenam-se todos os pontos de cada linha radial em termos dos valores de $M(x,y)$, armazenando os três valores mais elevados e as respectivas coordenadas dos pontos onde se registaram esses valores;
- Determina-se um grau de confiança, $R(x,y)$, de cada um dos pontos guardados. O valor mais elevado corresponde ao ponto do contorno da coroa.

O cálculo do grau de confiança é determinado de acordo com a seguinte expressão

$$R(x,y) = \begin{cases} e^{-\alpha(M(x,y)-\overline{M})^2} & \text{se } M(x,y) < \overline{M} \\ 1 & \text{se } M(x,y) \geq \overline{M} \end{cases} \tag{3}$$

onde, α é uma constante que impede que $R(x,y)$ seja muito pequeno, e $\overline{M}$ o valor médio dos três valores $M(x,y)$ entretanto guardados.



(a) Inicialização do Sistema (b) Gradiente

(c) Imagem auxiliar (d) Varrimento radial

Figura 1 - Passos iniciais da detecção de contornos

A detecção do contorno da raiz é efectuada a partir do contorno encontrado para a coroa, e obedece aos seguintes pontos:

- O primeiro ponto da raíz do lado direito/esquerdo corresponde ao último ponto do contorno da coroa do lado direito/esquerdo respectivamente;
- Os novos pontos do contorno da raíz são determinados tendo em conta o último ponto a ser encontrado e o seu "contexto". A medida do "contexto"é determinada pelos atributos I_{inner} e I_{outer} da imagem onde, I_{inner} é a intensidade média dos pixeis de uma pequena região da parte interna do dente e I_{outer} é a intensidade média dos pixeis de uma pequena região da parte externa do dente. Ver fig. 2

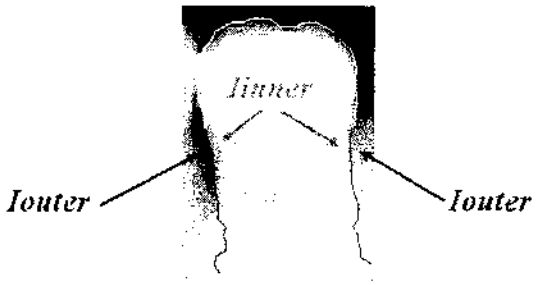


Figura 2 - Atributos do dente I_{inner} e I_{outer}.

Se o ponto i do contorno tiver coordenadas (x_i, y_i) as coordenadas do ponto $i+1$ são calculadas de acordo com a

iteração:

$$\begin{aligned} x_{i+1} &= \max_{x_i - r \leq x \leq x_i + r} (I_{inner} - I_{outer}) \\ y_{i+1} &= y_i + h \end{aligned} \tag{4}$$

onde, h corresponde ao incremento vertical, e r ao raio de procura horizontal.

B. Método da Probabilidade

Segundo Jain, et al. [1], também neste método, a extracção dos contornos do dente é constituída por duas fases, a detecção da coroa e a detecção da raíz. Observando as as radiografias dentárias, verifica-se que a região da imagem que contém a coroa é constituída por dois tipos de pixeis, os pixeis do dente, w_t e os pixeis do fundo da imagem, w_b. Denotando a intensidade dos pixeis por I, é, para este médtodo, importante obter uma estimativa credível da função densidade de probabilidade, $p(I)$. Segundo o autor, a estimativa pode ser efectuada por um de dois métodos. O método das janelas de Parzen [8] ou o método da mistura dos dois componentes. Usando o método da mistura dos dois componentes, pode-se descrever a função densidade de probabilidade através da seguinte expressão:

$$p(I) = p(I|w_b)P(w_b) + p(I|w_t)P(w_t) \tag{5}$$

onde $p(I|w_b)$ é a probabilidade de I condicionada pela distribuição de pixeis de fundo w_b e $p(I|w_t)$ é a probabilidade de I condicionada pela distribuição de pixeis representando os dentes. Na fig. 3, está representado o gráfico de $p(I)$ para uma dada imagem ou região de interesse. Analisando um conjunto representativo de imagens, verificou-se que os pixeis do fundo são, naturaltmente, os que têm os valores de intensidade mais baixos. Admitindo modelos probabilísticos gaussianos, pode assumir-se que $p(I|w_b$ corresponde ao primeiro componente do gráfico de $p(I)$. Determinando o primeiro componente, o segundo fica automaticamente determinado.

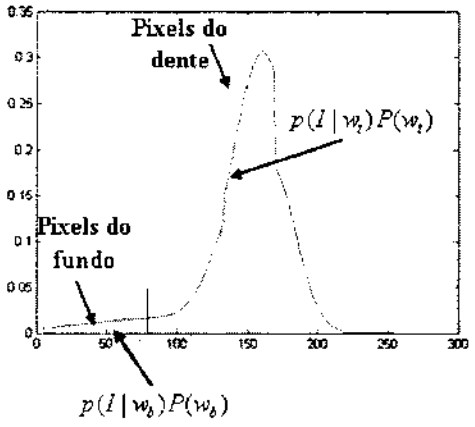


Figura 3 - Estimativa da função densidade de probabilidade das intensidades dos pixeis, $p(I)$.

De acordo com a regra de Bayes [9], a probabilidade posterior do pixel de intensidade I, se sabendo que se trata dum pixel do fundo da imagem, é dada pela seguinte expressão:

$$p(w_b|I) = \frac{p(I|w_b)P(w_b)}{p(I)} \tag{6}$$

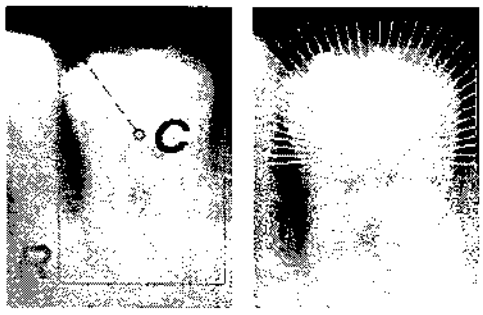
Identificando $p(I|w_b)P(w_b)$ e $p(I)$, determina-se facilmente $p(w_b|I)$. Como numa imagem do dente só existem duas classes de pixeis, $p(w_t|I)$ pode determinar-se por $p(w_t|I) = 1 - p(w_b|I)$. O conhecimento das probabilidades das distribuição dos pixeis, permite detectar todos os pontos do contorno da coroa, usando o seguinte algoritmo:

- Partindo do centro da coroa, efectua-se um varrimento radial para $0 \leq \theta \leq \pi$ com incremento $\Delta\theta$ (que determina o grau de precisão da detecção). Ver Fig. 4 (b);
- Para cada ponto P de cada linha radial, define-se P_{inner} e P_{outer}, como os pontos da vizinhança de P. Na Fig. 4 (c) está esquematizada a posição relativa destes pontos da vizinhança
- A probabilidade que o ponto P tem de ser um ponto do contorno é definida por:

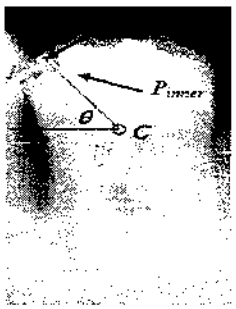
$$p(E) = p(w_b|I_{outer})p(w_t|I_{inner}) \tag{7}$$

onde, I_{inner} e I_{outer} são as intensidades de P_{inner} e P_{outer}, respectivamente.
- O ponto onde a probabilidade $p(E)$ for máxima corresponde ao ponto do contorno
- O processo repete-se para todas as linhas radiais, de forma a determinar todos os pontos do contorno

O contorno da raíz é determinado através do algoritmo usado para detectar a raíz no método do Gradiente.



(a) Inicialização do Sistema (b) Varrimento radial



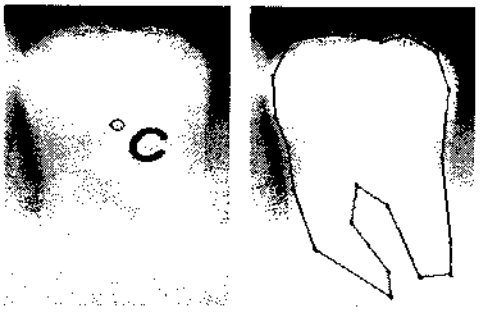
(c) Definição de Vizinhança do ponto P

Figura 4 - Método da probabilidade: inicialização

C. Modelos Deformáveis (Snakes)

Os métodos de segmentação que se baseiam me modelos deformáveis são numerosos e são muitas vezes ajustados para uma particular tarefa. No nosso caso optou-se por recorrer ao método clássico equipado por uma força

externa associada ao vector de fluxo do gradiente. Este método, descrito por Xu et al. [5]-[7], é usado para detectar o contorno do dente na sua totalidade. Como referido anteriormente, as radiografias dentárias não apresentam uma definição óptima. Assim, os dentes nelas representados apresentam várias alterações na sua morfologia, o que torna difícil o desenvolvimento de um sistema de segmentação totalmente automático. De modo a facilitar o processo de extracção dos contornos do dente, é necessário que o utilizador marque aproximadamente o centro da coroa, C, assim como, a forma do contorno a determinar. Partindo desta inicialização, o sistema detecta automaticamente os contornos do dente assinalado. Na fig. 5 está ilustrado um exemplo da inicialização deste método.



(a) Definição do centro da coroa (b) Definição do contorno inicial

Figura 5 - Método dos modelos deformáveis: inicialização

O valor do vector de fluxo do gradiente $g(x,y) = [u(x,y), v(x,y)]$ é aquele que minimiza a função da energia da imagem expressa pela equação

$$\varepsilon = \iint \mu.(u_x^2 + u_y^2 + v_x^2 + v_y^2) + |\nabla E|^2 |g - \nabla E|^2 \, dxdy \tag{8}$$

onde, ∇E é o vector gradiente normal às arestas detectadas; $u_x = \partial u/\partial x, u_y = \partial u/\partial y, v_x = \partial v/\partial x, v_y = \partial v/\partial y$; e μ é um parâmetro de regularização que sustenta a relação entre o termo em x e o termo em y. Esta constante depende da quantidade de ruído presente na imagem, deve assumir valores elevados para imagens com muito ruído, e valores baixos para imagens normais com menos ruído. Neste estudo usou-se $\mu = 0.1$.

Segundo Kass et al. [10], um contorno X é definido por $X(s) = [x(s), y(s)]$ com $s \in [0,1]$ e a função da energia num ponto $X(s)$ é dada por

$$E = \int_0^1 \frac{1}{2} \left[\alpha \left| X'(s) \right|^2 + \beta \left| X''(s) \right|^2 \right] + E_{ext}(X(s)) ds \tag{9}$$

onde, α representa a elasticidade ou tensão; β representa a rigidez; e X', X'' representam a 1ª e 2ª derivadas de $X(s)$, respectivamente. A energia externa E_{ext} é obtida a partir da imagem, tomando valores pequenos nas zonas dos contornos dos objecto. A curva dos Snakes que minimiza E, tem de satisfazer a equação de Euler:

$$\alpha X''(s) - \beta X''''(s) - \nabla E_{ext} = 0 \tag{10}$$

que, por sua vez, pode ser traduzida numa equação de equilíbrio de forças

$$F_{\text{int}} + F_{ext}^{(p)} = 0 \tag{11}$$

Em cada nova iteração, a curva dos *Snakes* é actualizada de modo a equilibrar as forças tendo em conta as características do meio (rigidez, elasticidade e força da imagem), contempladas por Xu, et al. [5]-[7]. A rigidez associa-se ao parâmetro α , e é responsável por impedir que o contorno curve em demasia (valores elevados fazem com que o contorno fique menos maleável). O valor usado no nosso trabalho foi 0. A Elasticidade associa-se ao parâmetro β , e é responsável por impedir que o contorno se alongue em demasia. O valor usado no nosso trabalho foi 0.05. A força Externa, é a força a que o contorno está sujeito por acção das características da imagem com destaque neste caso para o fluxo do vector gradiente. É também usual associar um parâmetro de controlo κ a esta força que no nosso caso foi 0.6. Usando este método, os contornos do dente são determinados como um todo exigindo pois um processamento posterior para separar o contorno nas suas partes constituintes, a coroa e a raíz. A exigência desta separação tem a ver com a necessidade de determinar a relação entre a coroa e a raíz, conforme adiante se verá.

III. INICIALIZAÇÃO DO SISTEMA

Dependendo do método escolhido para a extracção dos contornos do dente, é necessário efectuar uma inicialização ao sistema. Inicialmente é necessário indicar o lado do dente, isto é, indicar se o dente pertence ao maxilar superior ou ao maxilar inferior. Depois pode-se optar por um de três modos de inicialização, o modo manual, o modo semi-automático ou o modo automático. Nos itens que se seguem estão descritos de forma sucinta os dados que é necessário introduzir em cada um dos modos, para cada método. Saliente-se que o processo de inicialização é idêntico para os métodos do gradiente e da probabilidade, diferindo ligeiramente no método dos modelos deformáveis.

A. Modo Manual

Nos métodos do gradiente e da probabilidade, a inicialização consiste em marcar o centro da coroa, marcar um rectângulo em torno do dente e marcar um raio de procura usado para o varrimento angular, no processo de detecção dos pontos do contorno da coroa. Assim, inicializar manualmente o sistema consiste em introduzir, de forma manual, todos os dados necessários (o centro da coroa, o rectângulo, e o raio). Nas figs. 1,2, 4 representa-se a informação necessária ao modo manual de inicialização nos métodos do Gradiente e da Probabilidade. Nos modelos deformáveis a inicialização consiste em marcar o centro da coroa e vários pontos que representem, de forma grosseira, o contorno do dente conforme se demonstra na figura 5.

B. Modo Semi-automático

Neste modo de inicialização, são carregados sobre a imagem os dados usados na última inicialização, bastando ape-

nas ajustar a posição desses dados à imagem actual. Assim, este modo de inicialização permite economizar tempo e trabalho na inicialização. Na fig. 6 está representada uma inicialização semi-automática para os métodos do Gradiente e da Probabilidade

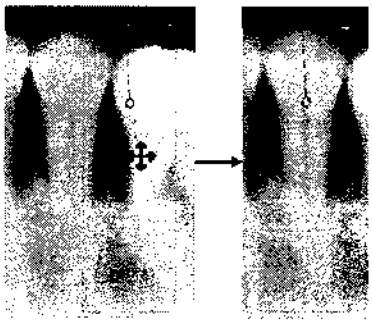


Figura 6 - Inicialização semi-automática nos métodos do Gradiente e da Probabilidade.

Na fig. 7 está representada uma inicialização semi-automática para os Modelos Deformáveis

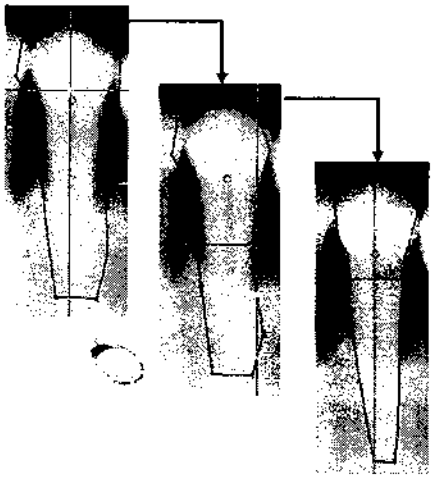


Figura 7 - Inicialização semi-automática nos Modelos Deformáveis.

C. *Modo Automático*

Interessa salientar que o modo automático de inicialização só pode ser usado para os métodos do Gradiente e da Probabilidade, uma vez que devido à natureza das imagens ainda não foi possível desenvolver um algoritmo capaz de permitir a inicialização automática para os Modelos Deformáveis. O modo automático é baseado no algoritmo descrito por Jain et al. [1]. Este algoritmo pode ser dividido em duas fases de execução: A detecção da zona que separa os dentes do maxilar de cima dos dentes do maxilar de baixo, e a separação dos vários dentes representados na imagem. Uma das técnicas mais eficazes, para determinar a zona de separação entre os dentes de cima e os de baixo, usa a projecção do histograma da imagem no eixo dos Y. Atendendo às características de uma radiografia dentária, a zona de separação entre os dentes de cima e os de baixo, deverá estar situada num nível onde a projecção do histograma sobre o eixo dos Y possuí um mínimo. Este mínimo justifica-se pelo elevado número de pixeis de fundo, nessa zona de transição, e que possuem uma intensidade menor que a intensidade dos pixeis do dente. Note-se no entanto, que existirão mais mínimos em toda a projecção do histograma, daí que seja necessária uma estimativa da posição da zona de separação. Esta estimativa será usada num cálculo probabilístico que validará o mínimo desejado, e que corresponde à zona de transição que se pretende determinar. Segue-se uma breve descrição do algoritmo usado na determinação da zona de transição.

- O utilizador tem de indicar a posição estimada para a zona de separação;
- Divide-se a imagem num determinado número de tiras verticais;
- Para cada tira da imagem são realizadas as seguintes operações:
  - Determina-se o somatório da intensidade dos pixeis, das colunas da tira, em cada linha da imagem. (Projecção do histograma no eixo dos Y's). Ver Fig. 8);
  - Guarda-se todos os mínimos da tira;
  - Determina-se a probabilidade de cada mínimo. A probabilidade de cada mínimo é determinada pela seguinte expressão:

$$p_{vi}(D_i, y_i) = p_{vi}(D_i)\, p_{vi}(y_i) \tag{12}$$

onde

$$p_{vi}(D_i) = c\left(1 - \frac{D_i}{\max_k D_k}\right) \tag{13}$$

$$p_{vi}(y_i) = \frac{1}{\sqrt{2\pi\sigma}} e^{-(y_i - y)^2/\sigma^2} \tag{14}$$

D_i corresponde ao valor do mínimo local, e $max_k D_k$ corresponde ao valor máximo da intensidade integrada.

  - O ponto onde a probabilidade for máxima corresponde ao nível da zona de separação nessa tira da imagem;
- Para finalizar o processo unem-se os níveis da zona de transição de cada tira da imagem, desenhando a zona que separa os dentes de baixo dos de cima; Ver fig. 8;

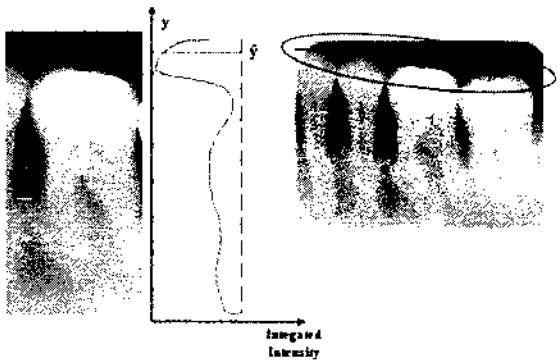


Figura 8 - Projecção do histograma numa de uma das tiras da imagem e consequente zona de separação detectada.

O método usado para separar os vários dentes da imagem é análogo ao método usado para separar os dentes de baixo

dos dentes de cima. Partindo da linha de separação encontrada, vão ser determinadas rectas perpendiculares a essa linha. Posteriormente, determina-se a intensidade integrada ao longo de cada uma dessas rectas. Fazendo uma análise global à intensidade integrada de todas as rectas, verifica-se a existência de vários mínimos locais. Cada um desse mínimos corresponde à recta de separação entre dentes. Ver fig. 9;

Figura 9 - Resultado da separação dos dentes.

Por vezes, os pontos do contorno da coroa podem assumir posições indesejadas. Estas situações podem ocorrer devido a vários factores, tais como: uma má inicialização, o ruído existente na imagem, entre outros de menor importância. De modo a tornar o sistema mais rigoroso e eficiente, implementámos um algoritmo que corrige as posições indesejadas. O algoritmo implementado pode ser descrito da seguinte forma:

- Enquanto existirem alterações a efectuar ou o número máximo de ajustes permitidos não for atingido:
  - Faz a estimativa da posição de cada ponto do contorno da coroa (Posição média entre o ponto imediatamente anterior e o ponto imediatamente posterior ao ponto);
  - Se a distância entre a posição real do ponto e a posição estimada exceder um determinado valor a posição desse ponto é substituída pela posição estimada e o sistema recomeça desde início a análise aos pontos do contorno. Caso contrário avança para o ponto seguinte.

O algoritmo também pode ser descrito através do fluxograma que está ilustrado na fig 10

Na fig 11 está ilustrado o resultado da validação dos pontos da coroa para um determinado dente.

IV. DETECÇÃO DO NÍVEL DA GENGIVA

As características morfológicas do dente que se pretendiam analisar exigia que se determinasse o nível da gengiva. Assim, depois de se ter detectado os contornos do dente desenvolveu-se um algoritmo com a função de detectar o nível da gengiva em cada um dos lados do dente (esquerdo ou direito). O algoritmo usado é semelhante ao que serviu para determinar o contorno da raiz nos métodos do Gradiente e da Probabilidade. Este algoritmo pode ser descrito pelos seguintes pontos:

- Desloca-se a parte esquerda e direita da raiz poucos pixeis (3, 4, 5, 6 dependendo das imagens) para a esquerda e para a direita respectivamente, de modo a assegurar que o varrimento do contorno da raíz é feito fora do dente mas que não interfira com os dentes vizinhos;
- Percorre-se a cada lado da raíz desde a fronteira com a coroa até à sua extremidade, analisando a vizinhança de cada ponto da raíz. O ponto onde a diferença entre I_{Gum} e I_{Outer} for máxima corresponde ao nível da gengiva, (Ver fig. 12). I_{Gum} é a intensidade média dos pixeis de uma pequena região da parte interna da gengiva. I_{Outer} é a intensidade média dos pixeis de uma pequena região da parte externa da gengiva.

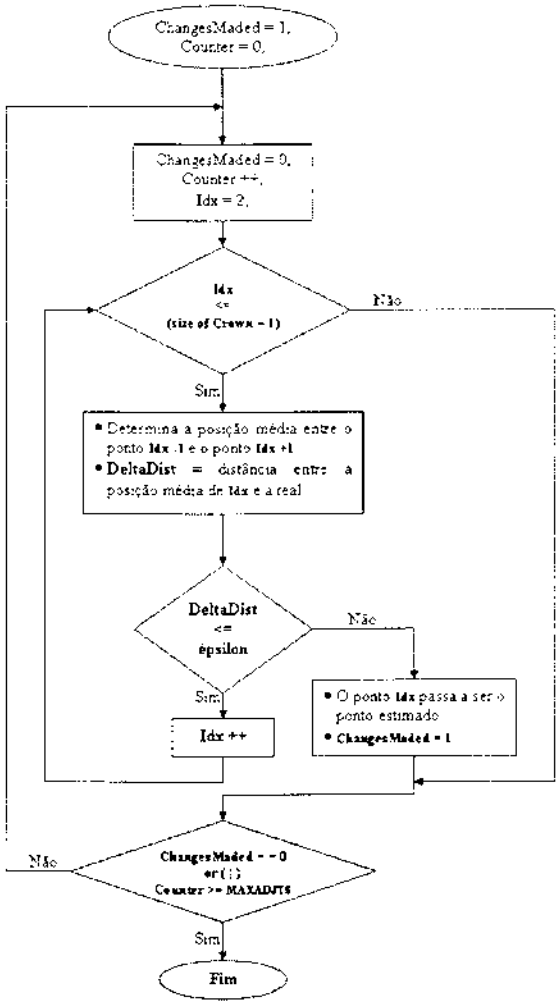


Figura 10 - Diagrama de fluxo do algoritmo de validação dos contornos da coroa.

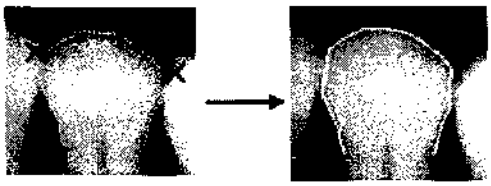


Figura 11 - Resultado do ajuste dos pontos da coroa.

O nível da gengiva pode ser expresso pela seguinte expressão:

$$GumLevel = \max_{y_i - r \leq y \leq y_i + r} (I_{Gum} - I_{Outer}) \qquad (15)$$

onde r representa o raio da vizinhança.

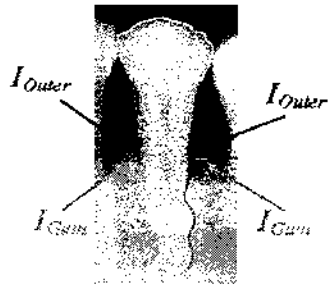
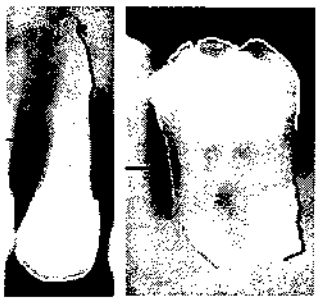


Figura 12 - Esquematização das propriedades I_{Gum} e I_{Outer}.

V. RESULTADOS EXPERIMENTAIS

Os métodos de detecção dos contornos foram testados e avaliados numa série de imagens provenientes de radiografias dentárias. O objectivo principal da extracção dos contornos é tornar possível a realização de um estudo das características morfológicas dos dentes, de modo a que possam ser avaliadas e detectadas eventuais patologias. Avaliando o desempenho do sistema nos vários métodos implementados, verifica-se que os resultados obtidos são bastante semelhantes entre eles, detectando com sucesso os contornos do dente. Na fig. 13 podem ser comparados os resultados de cada método para um conjunto de 2 dentes.

(a) Resultado do método do Gradiente (b) Resultado do método das Probabilidades



(c) Resultado dos Modelos Deformáveis;

Figura 13 - Resultados da segmentação

No que diz respeito ao tempo dispendido por cada um dos métodos, verifica-se que, o método do Gradiente e o método dos Modelos Deformáveis necessitam em média entre 15 a 17 segundos para identificar o contorno de cada dente. O método da Probabilidade é muito mais rápido, necessitando de menos de 5 segundos para detectar o mesmo contorno. A detecção dos contornos dos dentes serviu para determinar as suas propriedades mais importantes. Criou-se uma janela de informação, onde para além de estar presente uma ampliação do dente segmentado e dos seus contornos, estão também os seguintes parâmetros:

- A razão (em %) entre a altura da coroa e a altura da raiz;
- A razão (em %) entre a altura da parte da raiz descoberta da gengiva e altura da raiz completa;
- A razão (em %) entre a altura da parte da raiz descoberta da gengiva e altura da parte da raiz coberta pela gengiva

Como a raiz do dente pode estar mais descoberta de um lado do que do outro, foram calculadas as razões para os dois lados do dente. Além disso, para cada uma das razões, foi calculada a média entre os dois lados do dente. Na figura 14 está representado um exemplo da janela de informação para um determinado dente.

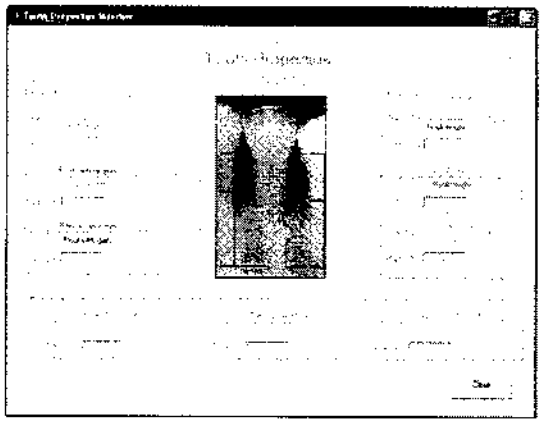


Figura 14 - Janela de visualização das propriedades do dente

VI. CONCLUSÕES E TRABALHO FUTURO

A título conclusivo, pode-se afirmar que é possível determinar os contornos de qualquer dente, em qualquer imagem proveniente de uma radiografia dentária, através de vários métodos. É legítimo afirmar que, partindo dos contornos detectados, é possível determinar várias características morfológicas do dente de modo a poder avaliar o estado de evolução de várias patologias do foro estomatológico. Depois de analisados os métodos implementados, verifica-se que todos apresentam bons resultados, fazendo com que o sistema desenvolvido corresponda às expectativas impostas no início do projecto. Futuramente este sistema pode ser integrado num software que possa ser usado no auxílio aos diagnósticos efectuados a partir de radiografias dentárias. Seria de toda a pertinência desenvolver uma base de dados de pacientes e respectiva interface, de modo a possibilitar o armazenamento das imagens e respectivos resultados obtidos da sua análise.

AGRADECIMENTOS

Agradece-se à Unidade de Investigação 127/94 IEETA da Universidade de Aveiro, o financiamento do projecto. Agradece-se ao Dr. Bruno G. Loos (Academic Center for Dentistry in Amsterdam) pelo disponibilidade das radiografias dentárias.

REFERÊNCIAS

[1] Anil K. Jain e Hong Chen, "Matching of dental x-ray images for human identification", *Pattern Recognition*, vol. 37, pp. 1519–1532, 2003.

[2] Anil K. Jain, Hong Chen, e Silviu Minut, "Dental biometrics: Human identification using dental radiographs", em *Proc. of 4th Int'l Conf. on Audio- and Video-Based Biometric Person Authentication (AVBPA)*, Guildford, UK, 2003, pp. 429 – 437.

[3] Hong Chen e Anil K. Jain, "Tooth contour extraction for matching dental radiographs", em *Proc. ICPR 2004*, Cambridge, UK, 2004, vol. III, pp. 522 – 525.

[4] H. Park, T. Schoepflin, e Y. Kim, "Active contour model with gradient directional information: Directional snake.", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11(2), pp. 252 – 256, 2001.

[5] C. Xu e J. L. Prince, "Gradient vector flow: A new external force for snakes", em *Proc. IEEE Conf. on Comp. Vis. Patt. Recog. (CVPR)*, Los Alamitos: Comp. Soc. Press, Ed., 1997, pp. 66 – 71.

[6] C. Xu e J. L. Prince, "Snakes, shapes, and gradient vector flow", *IEEE Transactions on Image Processing*, vol. 7(3), pp. 359 – 369, 1998.

[7] C. Xu e J. L. Prince, "Gradient vector flow deformable models", em *Handbook of Medical Imaging*, Isaac Bankman, Ed. Academic Press, 2000.

[8] R. O. Duda, P. E. Hart, e D. G. Stork, "Pattern classification, 2nd edition", em *Pattern Classification*, Wiley Interscience, Ed., pp. 164 – 174, New York, 2nd edition edição, 2001, Chapter 10.

[9] "An introduction to bayes rule", Relatório técnico, http://www.cim.mcgill.ca/ friggi/bayes/bayesrule/, 2004.

[10] M. Kass, A. Witkin, e D. Terzopoulos, "Snakes: Active contour models", *Int. Journal of Computer Vision*, vol. 1, pp. 321 – 331, 1987.

Avaliação da Qualidade de Modelos Poligonais dos Pulmões: Uma Experiência Controlada com Utilizadores

Samuel Silva[1], Joaquim Madeira, Beatriz Sousa Santos, Carlos Ferreira*

*Departamento de Economia, Gestão e Engenharia Industrial – Universidade de Aveiro

Resumo – **Malhas poligonais são muito usadas, em diferentes áreas, para representar a superfície de modelos: por exemplo, em Visualização Médica, permitem a representação de modelos de órgãos construídos a partir de dados de TAC. Quando essas malhas têm um número elevado de vértices e faces, pode não ser possível a manipulação interactiva dos modelos, sendo necessário aplicar um processo de simplificação para gerar uma sua versão menos complexa. Apesar de, nos últimos anos, terem sido desenvolvidos vários algoritmos de simplificação, é reduzido o número de trabalhos publicados comparando, para um mesmo modelo original e quanto à qualidade percebida pelos utilizadores, as malhas resultantes de diferentes processos de simplificação.**

Este artigo descreve a preparação, a execução e alguns resultados de uma experiência controlada com utilizadores para comparação dos modelos obtidos pela aplicação, a modelos pulmonares, de três métodos de simplificação de malhas triangulares. Pretendia-se determinar se, para dois níveis de simplificação, os modelos resultantes de algum desses métodos apresentam aos utilizadores uma melhor qualidade relativa.

Palavras chave – **Experiência controlada, Malhas triangulares, Simplificação, Qualidade.**

I. Introdução

Malhas poligonais são habitualmente usadas, em diferentes áreas de aplicação (Visualização Médica [1], Indústria Automóvel [2], Herança Cultural [3]), para representar a superfície de modelos. Em alguns casos, quando as malhas são definidas por um número elevado de vértices e faces, podem ocorrer restrições à utilização desses modelos quer em termos de manipulação interactiva, por o tempo necessário para o *rendering* ser significativo, quer para a sua transmissão via rede, pelo volume de dados envolvido, quer mesmo para o seu armazenamento em dispositivos com menor espaço de memória, como um *PDA*. Uma possível solução para estes problemas passa pela aplicação de um processo de simplificação a essas malhas poligonais, sendo gerada uma sua versão menos complexa, i.e., com menor número de vértices e faces.

Vários algoritmos de simplificação de malhas poligonais têm vindo a ser publicados no últimos anos [4], e foram definidos alguns índices de qualidade que permitem quer aferir as características de uma malha (p.ex., suavidade), quer comparar uma malha poligonal simplificada com a correspondente malha original [5]. Mas, apesar da crescente importância dada aos aspectos de qualidade e percepção em Computação Gráfica e Visualização [6], é reduzido o número de trabalhos publicados [7] comparando, para um mesmo modelo original, e quanto à qualidade percebida pelos utilizadores, as malhas resultantes de diferentes processos de simplificação. No entanto, esta é uma característica fundamental a ter em conta aquando da escolha do método de simplificação a usar.

Para comparar, relativamente à *"qualidade percebida"* e para vários níveis de simplificação, os modelos obtidos pela aplicação de métodos de simplificação a um conjunto de modelos originais, pode ser efectuada uma experiência controlada com utilizadores [8]. Esse procedimento compreende, inicialmente, o estabelecimento de uma hipótese e de um conjunto de variáveis a medir, a definição de uma metodologia e de um protocolo experimentais, a escolha de um grupo de participantes, e a selecção dos dados a serem recolhidos. Depois, é necessário aplicar o protocolo experimental e, posteriormente, proceder à análise estatística dos resultados, escolhendo os métodos apropriados.

Nas secções seguintes são detalhados os objectivos e as especificações da experiência planeada, é descrito o *software* desenvolvido para aplicação do protocolo experimental, são referidos os métodos estatísticos usados e apresentados alguns dos resultados obtidos.

II. Objectivos

Os objectivos principais do trabalho foram o estabelecimento de um protocolo experimental que permitisse a comparação de métodos de simplificação de modelos definidos por malhas poligonais, com base na *"qualidade percebida"* dos modelos simplificados, e a sua utilização para comparação de três métodos de simplificação num contexto em que a interacção com os modelos é fundamental, e suportada por *software* desenvolvido para esse fim. De início foi estabelecida uma hipótese simples: os modelos resultantes da aplicação de diferentes métodos de simplificação produzirão um efeito diferente nos observadores que, possivelmente, variará também com o nível (ou taxa) de simplificação. A primeira dificuldade encontrada prendeu-se com o planeamento da experiência: como poderia ser avaliado esse efeito nos observadores? Após algumas considerações, foi decidido usar, para esse fim, quer a *preferência* quer a *classificação* (*"rating"*) atribuídas pelos observadores aos modelos simplificados. Estas têm sido usadas nas ciências experimentais para obter a avaliação

[1] Financiado pela Unidade de Investigação 127/94 IEETA da Universidade de Aveiro

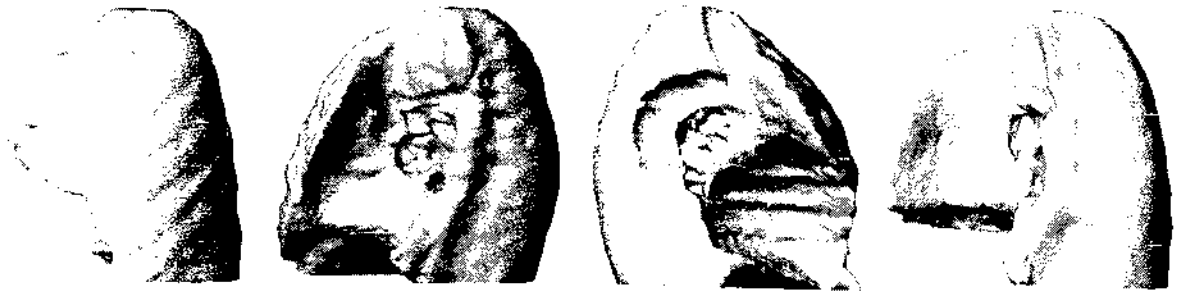


Figura 1 - Os quatro modelos pulmonares originais

relativa por parte dos participantes em experiências deste tipo [7]. Através de uma *classificação* os observadores associam a cada estímulo (i.e., modelo) um valor num dado intervalo e com um significado preciso. Com a indicação de uma *preferência* os observadores escolhem simplesmente o estímulo (i.e., o modelo) a que atribuem uma maior qualidade. Ambas representam decisões conscientes, e ambas demonstraram a sua utilidade em várias situações. Além do mais, a ordem de preferência e classificação são provavelmente os índices de fidelidade, i.e., semelhança visual com um original, mais adequados em processos de simplificação [7]. Foi também decidido registar os tempos de decisão e o número de interacções (efectuadas para cada modelo), dado que pareciam estar relacionados com o grau de dificuldade encontrado pelos observadores durante o estabelecimento da ordem de preferência e a atribuição de classificações aos modelos simplificados.

Considerou-se também que poderiam ser eventualmente obtidas algumas conclusões gerais úteis, a partir da comparação dos resultados dos vários métodos de simplificação, e mesmo verificado se o efeito nos observadores resulta não apenas do método de simplificação considerado mas também de outros factores, como, por exemplo, o nível de simplificação. Assim, além da aplicação de três métodos de simplificação distintos, foi decidido usar dois níveis de simplificação (20% e 50%).

Na experiência foram usados modelos de pulmões, inicialmente gerados a partir de contornos segmentados de exames de TAC [9], usando um simples algoritmo de reconstrução e a estrutura de dados, para armazenamento de malhas poligonais, fornecida pela biblioteca *OpenMesh* [10]. Estes modelos originais foram, depois, simplificados usando três métodos distintos: dois deles são disponibilizados pela *OpenMesh* e usam quádricas de erro [11], ou quádricas de erro e a variação dos vectores normais às faces (*"normal flipping"*) como critérios de simplificação; o terceiro (*QSlim* [11]) é também um método de simplificação baseado em quádricas de erro, sendo muito usado e bastante referido na literatura [7].

III. EXPERIÊNCIA

Aquando do planeamento de uma experência controlada, é necessário estabelecer as suas características principais [12]: hipótese e variáveis, participantes, conjunto de dados de teste, metodologia experimental, protocolo, conjunto de dados recolhidos e os métodos estatísticos a utilizar. Estes aspectos são descritos de seguida, no contexto da experiência realizada.

A. Hipótese

A hipótese é uma previsão do resultado da experiência: é estabelecida em termos de variáveis, indicando se uma variação numa variável independente afectará (ou não) uma variável dependente. O objectivo da experiência é mostrar se esta previsão é correcta: isto é realizado demonstrando (ou não) que a hipótese nula, que afirma que alterações nas variáveis independentes não conduzem a alterações das variáveis dependentes, não se verifica.

No caso da nossa experiência, pretende-se mostrar especificamente se o método de simplificação e o nível de simplificação das malhas poligonais tiveram algum efeito nas respostas dos participantes, i.e., influenciaram as suas preferências e classificações quanto à qualidade dos modelos, os tempos de decisão e o número de interacções efectuadas.

B. Variáveis

De acordo com a hipótese anterior, foram escolhidas as seguintes variáveis:

- variáveis independentes: método de simplificação (com três níveis: *QSlim*, *OpenMesh* e *OpenMesh* com *"normal flipping"*) e nível de simplificação (com dois níveis: 20% e 50% do número de faces dos modelos originais);
- variáveis dependentes: preferências; classificações; tempos de decisão; número de interacções.

Será desejável que cada modificação no valor de uma variável dependente se deva a uma modificação de uma variável independente. No entanto, outras variáveis, como o sexo, a idade e a experiência prévia dos participantes na manipulação de modelos de objectos 3D, poderão influenciar os resultados. Assim, esta informação foi usada para caracterizar o perfil de cada participante.

C. Participantes

O grupo de participantes foi composto por 32 estudantes: 18 da área de Engenharia e 14 de Radiologia, 20 homens e 12 mulheres, cujas idades variavam entre 18 e 55 anos; no entanto a maioria (21) tinha entre 18 e 25 anos de idade. Vinte dos participantes declararam ter alguma experiência na visualização/manipulação de modelos 3D.

Note-se que se pretendia também verificar se a familiaridade, que os estudantes de Radiologia têm com a

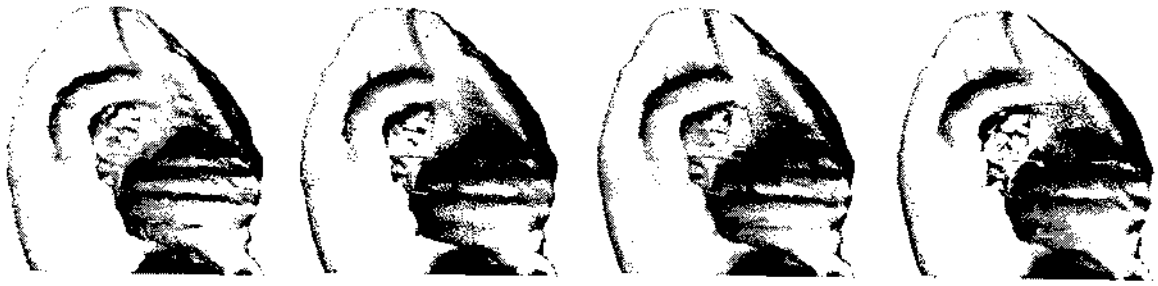


Figura 2 - Exemplo de um conjunto de modelos de teste

visualização de imagens, poderia influenciar o seu desempenho nesta experiência.

D. Conjuntos de Modelos de Teste

Os conjuntos de modelos de teste foram gerados a partir de um grupo de quatro modelos pulmonares originais (figura 1), construídos a partir de contornos segmentados em imagens de TAC. A tabela I apresenta o número de vértices e de faces definindo esses quatro modelos originais.

Para cada um dos modelos originais, foram estabelecidos dois níveis de simplificação: 20% e 50% do número original de faces. Para cada nível de simplificação, foram gerados três modelos, usando três processos diferentes: método *QSlim*, método de simplificação da biblioteca *OpenMesh*, e este último método usando também critérios de *"normal flipping"*. Para a utilização de *"normal flipping"*, foi previamente determinado o valor mínimo de variação angular, dos vectores normais associados a cada vértice de uma malha, necessário para a obtenção do nível de simplificação desejado, dado que a utilização habitual deste critério consiste em estabelecer a máxima variação admissível, e não o número de faces que se deseja obter no modelo simplificado.

Deste modo, foram obtidos oito conjuntos de modelos de teste, cada um deles contendo um modelo original e três versões simplificadas, para o mesmo nível de simplificação. A figura 2 apresenta um dos conjuntos de modelos de teste.

Tabela I
NÚMERO DE VÉRTICES E DE FACES DOS MODELOS ORIGINAIS

Modelo	# Vértices	# Faces
A - Pulmão 1	9.367	18.609
B - Pulmão 2	9.740	19.334
C - Pulmão 3	8.948	17.747
D - Pulmão 4	4.811	9.514

E. Metodologia Experimental

Foi escolhida uma metodologia experimental *"dentro de grupos"* (ou *"repeated measures"*), na qual cada participante é confrontado com cada um dos diferentes estímulos (i.e., modelos). Esta metodologia pode sofrer, para um mesmo participante, do efeito de transferência de aprendizagem, mas este pode ser atenuado se a ordem pela qual os estímulos são apresentados variar para cada um dos participantes. Esta metodologia é menos onerosa que uma metodologia *"entre grupos"* (na qual a cada participante é apenas apresentado um estímulo), dado que é necessário um número mais reduzido de participantes e há também uma menor probabilidade da ocorrência de variações significativas entre participantes.

Atendendo à metodologia escolhida, e à possibilidade de os resultados experimentais serem influenciados por algumas variáveis externas (p.ex., efeitos de aprendizagem, nervosismo na primeira tarefa ou cansaço na última), foi decidido: primeiro, apresentar de modo aleatório, a cada participante, os conjuntos de modelos de teste e, segundo, para cada participante, a ordem de apresentação dos modelos de cada conjunto foi também escolhida aleatoriamente.

Há algumas condições externas que podem influenciar os resultados da experiência: condições de visualização, vistas pré-definidas dos modelos, etc. Para minimizar a sua influência, foram feitas as seguintes escolhas:

- Todos os modelos foram representados usando uma projecção paralela ortogonal e o método de sombreamento de Gouraud.
- Cada participante poderia manipular os modelos, escolhendo a sua orientação e tamanho. Esta é considerada uma característica muito importante desta experiência particular, já que, quando se comparam modelos 3D, deverá ser permitida a sua comparação usando qualquer direcção de observação, orientação ou factor de escala considerados importantes pelo observador, e não apenas uma só ou um número muito reduzido de vistas pré-definidas.
- Durante a experiência, as condições de visualização (tamanho do ecrã, distância ao ecrã, iluminação da sala, etc.) foram semelhantes para todos os participantes.

Em resumo, trata-se de um *design* experimental 2 × 3: são usados dois níveis e três métodos de simplificação, e cada um dos participantes foi confrontado com cada um dos modelos em todas as condições experimentais, embora por uma ordem diferente.

F. Protocolo

De início, foi dada a todos os participantes uma explicação sobre o contexto e os objectivos da experiência, bem como das tarefas que teriam de executar; depois foi-lhes pedida alguma informação caracterizando o seu perfil. A experiência, que foi dividida em duas fases distintas, iniciou-se de seguida. Note-se que, no início de cada fase, foram

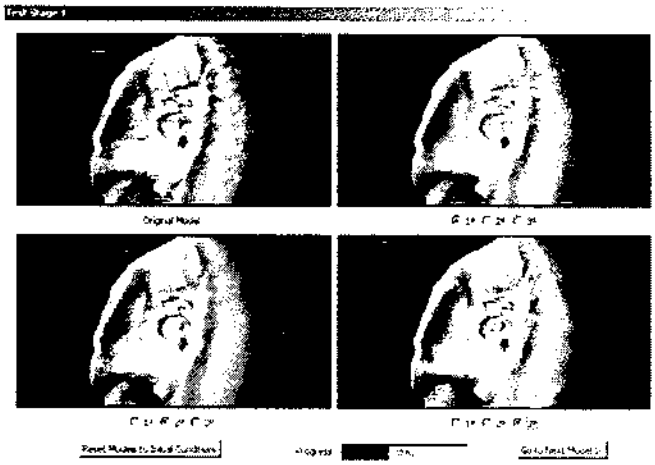


Figura 3 - A interface da primeira fase da experiência: um conjunto de modelos de teste (modelo original à esquerda em cima)

apresentados aos participantes conjuntos de modelos[1] para treino: puderam assim familiarizar-se com as tarefas e as operações de manipulação de modelos.

Na primeira fase (figura 3), foi sucessivamente apresentado, a cada participante, cada um dos oito conjuntos de modelos de teste, composto por quatro modelos: um modelo original e três suas versões simplificadas, correspondendo à aplicação dos três métodos, para um dado nível de simplificação. Cada participante tinha de ordenar os três modelos simplificados, de acordo com a qualidade relativa que atribuia a cada um. Apesar de ser pedido aos participantes que tentassem atribuir três classificações distintas (primeiro, segundo e terceiro lugar), permitiam-se situações de empate, para acolher casos em que a distinção entre modelos se revelasse muito difícil.

Na segunda fase (figura 4), foi sucessivamente apresentado, a cada participante, um par de modelos constituído por um dos modelos originais e por uma sua versão simplificada, retirada de um dos oito conjuntos de modelos de teste. Cada participante tinha de classificar usando uma escala de Likert [13] [1 (muito mau) a 5 (muito bom)] o modelo simplificado, por comparação com o original, novamente com base na qualidade percebida.

É de notar que, durante ambas as fases da experiência, os participantes puderam manipular os modelos, escolhendo a sua orientação e tamanho, existindo um mecanismo de sincronização que impedia a existência de discrepâncias (p.ex., de orientação) entre os modelos apresentados em cada instante. A aplicação desenvolvida para suportar este protocolo experimental será descrita na próxima secção.

G. Informação recolhida

De modo a estabelecer o perfil de cada participante, foi recolhida informação quanto ao sexo, à idade ($]18, 25]$; $]26, 35]$; $]36, 55]$; > 55) e à sua familiaridade com a visualização e manipulação de modelos 3D.

Na primeira fase, foram registadas as preferências dos participantes (modelos simplificados em primeiro, segundo e terceiro lugares), a ordem de apresentação dos conjuntos de modelos, o número de interacções efectuadas com os modelos, bem como o tempo de decisão. Na segunda fase, foram registadas as classificações atribuídas a cada modelo simplificado, a ordem da sua apresentação, bem como o número de interacções efectuadas e o tempo necessário para a classificação.

Após a experiência, foi dada a oportunidade aos participantes de efectuar alguns comentários e relatar eventuais dificuldades experimentadas.

[1] Na fase de treino foram usados modelos pulmonares distintos dos modelos de teste.

IV. APLICAÇÃO DESENVOLVIDA

Para realização do protocolo experimental definido, foi desenvolvida uma aplicação, usando *Visual C++* e baseada nas bibliotecas *OpenMesh*, para definição e armazenamento das malhas poligonais, e *OpenGL*, para a visualização e o *rendering* dos modelos. Foi também usada a *Fox Toolkit* [14] para definição da interface e gestão da interacção com os utilizadores. Nesta secção são apresentadas as principais características da aplicação desenvolvida.

A. Leitura, Armazenamento e Visualização de Modelos

Todos os modelos poligonais usados na experiência se encontram armazenados em disco, usando o formato OBJ, e são lidos usando as funcionalidades da biblioteca *OpenMesh*. Esta fornece uma estrutura de dados que permite facilmente representar as malhas poligonais em memória, e aceder aos seus componentes (vértices, arestas e faces) para efeitos de *rendering*. Os modelos são todos lidos de disco antes do início da experiência, de modo a reduzir os tempos de espera entre as diferentes etapas da experiência. A biblioteca *OpenGL* é usada para suportar a visualização dos modelos: para um melhor desempenho são construídas listas de desenho (*"display lists"*) para cada modelo.

B. Interface

A *Fox Toolkit* apresenta um *widget* que permite a visualização e a manipulação de uma cena criada usando *OpenGL*. Este *widget* foi usado na aplicação para apresentar os modelos ao utilizador e para fornecer as funcionalidades de manipulação (*zoom*, *pan* e rotação).

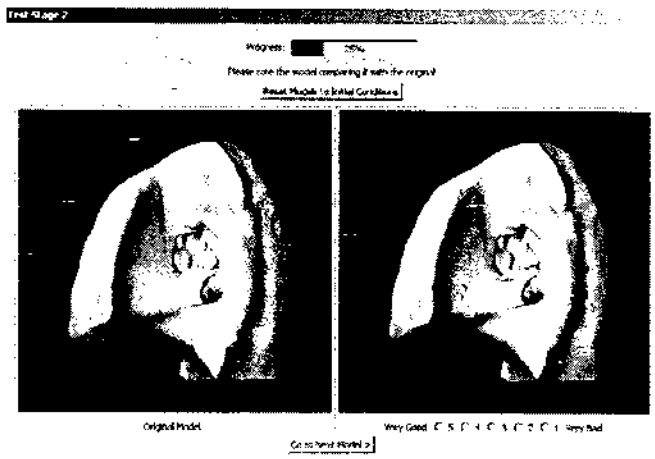


Figura 4 - A interface da segunda fase da experiência: modelo original (à esquerda) e modelo simplificado

A interacção com os modelos possui uma característica particular: sempre que o utilizador aplica uma operação a um dos modelos, esta é também internamente aplicada a todos os outros modelos visíveis, mantendo-se assim os modelos sincronizados para evitar qualquer discrepância que possa desorientar o utilizador. Note-se que só o modelo directamente manipulado pelo utilizador é alterado em tempo-real, sendo a sincronização feita apenas no fim da operação pretendida, de modo a minimizar o esforço computacional e a permitir uma interacção mais suave. Na interface existe um botão que permite repor todos os modelos nas condições iniciais de posição, orientação e tamanho.

Houve o cuidado de fornecer informação ao utilizador quanto ao desenrolar da experiência. Assim, sempre que uma operação mais demorada (p.ex., o carregamento dos modelos) é executada, apresenta-se ao utilizador uma barra de progresso, com a indicação de qual tarefa está a ser realizada e de quanto falta para o seu fim. Para minimizar a ocorrência de erros, o utilizador só pode passar à tarefa seguinte depois de dar uma resposta completa: atribuir uma ordem aos três modelos simplificados de cada conjunto de teste (primeira fase) ou uma classificação a cada modelo simplificado (segunda fase).

C. *Registo de dados*

Para cada participante na experiência, os dados registados em cada fase são armazenados em ficheiro e, após o fim da experiência, enviados automaticamente para um servidor. Assim, evita-se uma recolha manual de todos os ficheiros. Os dados são armazenados num formato textual, permitindo a sua fácil verificação, se necessário.

Foi também desenvolvida uma simples aplicação adicional para fazer o *parsing* dos ficheiros de dados de todas as experiências e reunir toda a informação num só ficheiro, com um formato que pode ser lido por uma folha de cálculo ou um programa de análise estatística de dados.

D. *Experiência-Piloto*

É de notar que foi previamente efectuada uma experiência-piloto de modo a testar o protocolo, o processo de registo dos dados da experiência e todo o *software* desenvolvido. Como resultado, foram feitos alguns melhoramentos: por exemplo, a adição de uma barra de progresso dando aos participantes uma indicação relativa de que percentagem da experiência já se encontra concluída. Isto revelou-se bastante importante, de modo particular na segunda fase da experiência, pois o grande número de situações de comparação e classificação de modelos que ocorre (há seis modelos simplificados para cada modelo original) induz em alguns utilizadores a sensação de que haverá um problema com a aplicação.

V. ANÁLISE DE RESULTADOS

O primeiro passo na análise dos resultados foi a realização de uma Análise Exploratória [15]. Obteve-se assim alguma informação geral sobre as relações estruturais, mostrando as amplitudes, assimetrias, localizações, *outliers*, etc. Foram também obtidas algumas indicações relativamente quer aos métodos estatísticos a usar para testar a hipótese original, quer mesmo a ideias sobre outras hipóteses. Os métodos estatísticos deverão ser os mais adequados para o tipo de resultados da experiência (tamanho, distribuição e natureza da amostra, escala de medida, etc.), e também para o tipo de tarefas executadas na experiência. Os dados registados relativamente à ordem de preferência e às classificações são ordinais; os tempos de decisão são quantitativos e o número de interacções é medido numa escala quantitativa mas discreta e, portanto, foram usados diferentes métodos adequados a cada tipo de dados. A análise foi efectuada usando o *software* STATISTICA [16].

Da análise de resultados foram obtidas algumas conclusões gerais: os alunos de Radiologia apresentam (em geral) menores tempos de decisão que os alunos de Engenharia, e as mulheres efectuaram um menor número de interacções com os modelos que os homens. De seguida apresentam-se os resultados principais associados a cada uma das fases da experiência. Uma descrição e uma análise mais detalhadas dos resultados obtidos é apresentada em [17].

A. *Primeira Fase — Preferências*

Para todos os modelos, os dados recolhidos relativos aos tempos de decisão e ao número de interacções foram agru-

pados de acordo com o nível de simplificação dos modelos. O nível de simplificação influenciou os tempos de decisão: os participantes decidiram mais depressa para modelos com um nível de simplificação de 20%.

Para o nível de simplificação de 20% os participantes preferiram os modelos simplificados usando o método *QSlim* (maior número de primeiros lugares), depois os modelos simplificados usando o método da *OpenMesh* e, em terceiro lugar, os modelos simplificados usando o método anterior com *"normal flipping"*. No entanto, esta tendência não é tão pronunciada para o nível de simplificação de 50%.

B. Segunda Fase — "Ratings"

Os resultados parecem indicar que o nível de simplificação tem uma influência real quer nos tempos de decisão, quer no número de interacções registadas. Para o nível de simplificação de 20% os observadores decidiram mais depressa e efectuaram menos operações sobre os modelos.

Os observadores parecem ser sensíveis ao nível de simplificação, atribuindo menor classificação aos modelos com nível de simplificação de 20%, em particular aos simplificados usando o método da *OpenMesh* com *"normal flipping"*. Para o nível de simplificação de 50% as classificações foram melhores, agora com ligeira vantagem para os modelos processados usando o método da *OpenMesh* com *"normal flipping"*.

VI. CONCLUSÕES E TRABALHO FUTURO

Foi descrita a preparação e a execução de uma experiência controlada com utilizadores para comparação dos modelos obtidos pela aplicação, a modelos pulmonares, de três métodos de simplificação de malhas triangulares.

Da análise dos dados recolhidos, parece poder concluir-se que os utilizadores são sensíveis quer ao método de simplificação usado, quer ao nível de simplificação dos modelos pulmonares. No entanto, deve salientar-se que os utilizadores parecem reagir de modo distinto para cada um dos dois níveis de simplificação: para 20%, o método *QSlim* parece ser o preferido, enquanto que, para 50%, é o método da biblioteca *OpenMesh* com *"normal flipping"* que obtém melhores resultados.

Apesar da metodologia empregue ter sido desenvolvida para esta experiência particular, é suficientemente genérica pra poder ser aplicada em outras experiências controladas de comparação de modelos 3D. Um dos objectivos de curto-prazo é a realização de uma experiência semelhante, com um maior número de participantes e usando modelos de diferentes tipos, para se tentar verificar se os resultados agora obtidos são generalizáveis, ou dependem da natureza dos modelos considerados. É também importante analisar se existe alguma correlação entre os resultados obtidos e alguns dos indíces de qualidade habitualmente usados para classificar malhas poligonais [5].

AGRADECIMENTOS

Os autores agradecem a todos os participantes na experiência e, também, ao Prof. Augusto Silva, por ter permitido quer a realização da experiência nas suas aulas, quer a participação dos seus estudantes de Radiologia.

O primeiro autor agradece à Unidade de Investigação 127/94 IEETA, da Universidade de Aveiro, a Bolsa de Iniciação à Investigação que vem financiando as suas actividades.

REFERÊNCIAS

[1] M. Jackowski, M. Satter, e A. Goshtasby, "Approximating digital 3D shapes by rational gaussian surfaces", *IEEE Trans. on Visualization and Computer Graphics*, vol. 9(1), pp. 56–69, 2003.

[2] G. Sußner, G. Greiner, e S. Augustiniack, "Interactive examination of surface quality on car bodies", *Computer-Aided Design*, vol. 36, pp. 425–436, 2004.

[3] G. Guidi, J. Beraldin, e C. Atzeni, "High-accuracy 3D modeling of cultural heritage: The digitizing of Donatello's "Maddalena"", *IEEE Trans. Image Processing*, vol. 13(1), pp. 370–380, 2004.

[4] D. Luebke, "A developer's survey of polygonal simplification algorithms", *IEEE Computer Graphics and Applications*, vol. 21(3), pp. 24–35, 2001.

[5] M. Roy, S. Foufou, e F. Truchetet, "Mesh comparison using attribute deviation metric", *Internation Journal of Image and Graphics*, vol. 4(1), pp. 1–14, 2004.

[6] C. O'Sullivan, S. Howlett, R. McDonnel, Y. Morvan, e K. O'Conor, "Perceptually adaptive graphics", em *EUROGRAPHICS 2004 - State of The Art Reports*, Grenoble, France, 2004, pp. 141–164.

[7] B. Watson, A. Friedman, e A. McGaffey, "Measuring and predicting visual fidelity", em *Proc. SIGGRAPH 2001*, 2001, pp. 213–220.

[8] J. Preece, Y. Rogers, H. Sharp, Benyon D., S. Holland, e T. Carey, *Human Computer Interaction*, Addison Wesley, 1994.

[9] A. Silva, J. S. Silva, B. S. Santos, e C. Ferreira, "Fast pulmonary contour extraction in X-ray CT images, a methodology and quality assessment", em *Proc. SPIE 2001 vol 4321, Progress in Biomedical Optics and Imaging*, San Diego, 2001, vol. 26, pp. 216–224.

[10] M. Botsch, S. Steinberg, S. Bischoff, e L. Kobbelt, "OpenMesh - a generic and efficient polygon mesh data structure", em *1st OpenSG Symp.*, Darmstadt, Alemanha, 2002.

[11] M. Garland e P. Heckbert, "Surface simplification using quadric error metrics", em *Proc. SIGGRAPH 1997*, 1997, pp. 209–216.

[12] A. Dix, J. Finlay, G. Abowd, e R. Beale, *Human Computer Interaction*, Prentice Hall, 3ª edição, 2004.

[13] V. Barnett, *Sample Survey Principles and Methods*, Arnold Hodder, 3ª edição, 2003.

[14] J. van der Zijp, "Fox toolkit", *http://www.fox-toolkit.org*, (online Mar/2005).

[15] D. Hoaglin, F. Mosteller, e J. Tukey, *Understanding Robust and Exploratory Data Analysis*, John Wiley & Sons, 1983.

[16] "STATISTICA 6.0", *http://www.statsoft.com*, (online Mar/2005).

[17] S. Silva, B. S. Santos, J. Madeira, e C. Ferreira, "Comparing three methods for simplifying mesh models of the lungs: an observer test to assess perceived quality", em *Proc. SPIE 2005 vol 5749, Image Perception, Observer Performance, and Technology Assessment (in press)*, San Diego, 2005.

Integração de Vídeo em Sistemas de Comunicação e Multimédia: taxonomia, paradigmas e boas práticas

André Valentim Almeida (*), Beatriz Sousa Santos , Carlos Ferreira (**)
(*) Universidade do Porto
(**) DEGEI, Universidade de Aveiro – CIO, Universidade de Lisboa

Resumo – **O presente artigo visa reflectir sobre a temática da integração de Vídeo em Sistemas de Comunicação e Multimédia. Tendo por base as dificuldades existentes neste domínio, é desenvolvida uma listagem de causas que podem estar na origem do problema. Posteriormente, são elaboradas duas taxonomias com o objectivo de caracterizar o Vídeo, tanto no contexto da integração em Multimédia, como também na sua vertente individual. Para finalizar, e com base em alguns dos pressupostos enunciados, são enunciadas boas práticas e paradigmas na produção e integração de Vídeo em Multimédia.**

Abstract - **The present work aims to discuss the integration of Video in Communication Systems and Multimedia. Having noticed several problems within this domain, we developed a list of possible causes responsible for the difficult integration. This exercise is followed by the framework for a Video taxonomy. Finally we sketch the itemization of good practices for the production, integration and distribution of Video in Multimedia systems.**

I. INTRODUÇÃO

É inquestionável, actualmente, a presença vincada do Multimédia e das Imagens em Movimento no mundo contemporâneo. Todavia, a anunciada integração do Vídeo na matriz do Multimédia – tido como um dos elementos constituintes da sua identidade – encontra-se longe do horizonte, prevalecendo, ainda, um desfasamento significativo entre os circuitos audiovisual e multimédia. Tomando a televisão e o computador pessoal como paradigmas, os esforços empreendidos no sentido da sua fusão – ainda que tecnologicamente exequíveis – têm redundado em fracassos comerciais [1]. Quando este cruzamento se efectiva, e através da análise da aplicação de vídeo em diversos cenários multimédia, constata-se que, na sua globalidade, não há evidente valor acrescentado na inclusão de *clips* de vídeo [2]. Esta situação evidencia a dificuldade no exercício de integração do Vídeo em Multimédia e, ao mesmo tempo, a carência de reflexão nesta área.

É, então, neste contexto que se enquadra o presente artigo, que pretende reflectir sobre os conceitos de Multimédia e Imagens em Movimento e dificuldades de integração inerentes (secção II), desenvolver uma taxonomia relativa à integração de Vídeo em Multimédia (secção III), expor algumas boas práticas e paradigmas na produção e integração de Vídeo para Multimédia (secção IV) e, finalmente, traçar as respectivas conclusões (secção V).

II. MULTIMÉDIA E IMAGENS EM MOVIMENTO

A literatura que tenta definir sumariamente o Multimédia é vasta (e, não raras vezes, lacónica e contraditória). Em publicação relativa à área, Simões e Pinto [3] definem Multimédia como a «(...) integração de elementos de natureza diferente no seio de um mesmo sistema ou de uma mesma plataforma». Fluckiger [4] estreita o conceito, afirmando que o Multimédia digital é o campo que se debruça sobre a integração controlada por computador de texto, gráficos, imagens e Imagens em Movimento, animação, som, e qualquer outro tipo de informação que possa ser representado, armazenado, transmitido e processado digitalmente, opinião partilhada por Chapman e Chapman [5]. No entanto, adverte para o facto de que o cruzamento de qualquer par destes elementos não é condição suficiente para ser legitimamente apelidado de Multimédia. Para o autor, é necessário que pelo menos um média discreto – texto, gráficos ou imagem – seja associado a um média contínuo – áudio ou vídeo – para que possa ser legitimamente considerado Multimédia. À luz desta definição, diversos produtos comummente considerados multimédia não o são, carecendo da integração de média contínuos. No caso específico do Vídeo, verifica-se que é aquele que, provavelmente, apresenta o maior fosso no contexto da integração. Esta situação desperta alguma estranheza, sobretudo quando na literatura, de maior ou menor pendor científico, o Vídeo é tido como um média portador de uma extenso espectro de vantagens.

Nos casos em que a inclusão do Vídeo em Multimédia se concretiza, a sua avaliação está longe de ser positiva. Lopes, Moreira *et al.* [6] reduzem esta acção a meras «operações de "juntar", sem preocupação de correlação entre os diversos componentes.». Lopes [7] vai mais longe: na avaliação de uma série de produtos multimédia constata que a grande maioria padece de «alguma falta de bom senso», traduzidos muitas vezes num «folclore de inadequações».

De facto, o Multimédia encontra-se ainda muito longe da perfeita integração dos elementos da sua matriz. O problema não é novo, uma vez que, citando um exemplo curioso, o próprio livro, sistema de complexidade substancialmente inferior, necessitou de algumas décadas até atingir a estrutura base que hoje lhe conhecemos.

Com base no exposto, e com apoio na literatura e experiência pessoal, enumeram-se, de seguida, algumas razões que podem estar na origem da difícil integração do Vídeo em Multimédia:

- As actuais ferramentas de suporte ao desenvolvimento de projectos multimédia não são adequadas às especificidades únicas daqueles que trabalham neste meio, provavelmente devido a uma falha na percepção das suas necessidades e, também, através da importação de modelos não adequados [8, 9]. Marks e Davis [9] defendem que algumas das actuais concepções de *design* de *software*, como o «*iteractive design*» e o «*concurrent design*», são, em grande parte, sistemas de desenvolvimento de *software* em contextos de não Multimédia;
- O quotidiano ainda está preso aos cânones do texto, da palavra escrita, tornando a ruptura deste esquema mental uma tarefa complexa [10]. Esta situação pode conduzir a uma sobreutilização do texto em Multimédia;
- Sobre o Multimédia paira, ainda, uma forte dualidade entre forma e conteúdo. «Na realidade, a sua utilização [Vídeo] é realizada mais como um *gadget*; utiliza-se porque o Vídeo digital em Multimédia é "mais giro" ou tem mais "piada".» [2]. A este facto não é alheia a euforia inerente ao período actual – o despontar da "revolução" multimédia –, fase geralmente pautada pela descoberta e experimentação. A preocupação com a estética é, por vezes, superior ao conteúdo, princípio desencontrado das necessidades dos utilizadores. A introdução num produto de um elemento da matriz Multimédia, como são o Vídeo e o Áudio, deverá ser realizada somente quando existir valor acrescentado;
- As escolas de produção Multimédia não caminham totalmente a par das de produção audiovisual. Como consequência dessa dessincronia resultam linguagens distintas, nem sempre compreendidas por ambas as partes. Em contexto de integração, é comum deparar com dois cenários distintos: tanto profissionais com formação Multimédia a trabalhar em Vídeo, conduzindo a vídeos de qualidade inferior, como, pelo contrário, bons vídeos realizados por profissionais da área, mas que pecam na sua integração com o produto Multimédia;
- A factura a pagar pelo trabalho em Vídeo com qualidade é ainda alta. Ainda que o preço do equipamento de vídeo tenha, em alguns sectores, reduzido drasticamente, será necessário optar por uma equipa especializada, que é, no mínimo, numerosa. Gleicher, RachelHeck *et al.* [11] consideram que «*merely capturing an event on video does not make the video watchable.*» Este facto é notado por Lopes, Moreira *et al.* [6], que afirmam: «(...) poucos são os que fazem ou vendem filmes, havendo claramente a distinção entre filmagens amadoras e o circuito profissional do audiovisual.»;
- O Vídeo é um elemento que consome bastante tempo na sua produção, condição que, em algumas situações, não se coaduna com a filosofia Multimédia, onde os prazos são, com frequência, muito apertados;
- Quanto à Internet, esta é, na sua grande maioria, um meio de grande volatilidade de conteúdos e de baixo custo, onde o vídeo tem dificuldade em penetrar. A necessidade quer de servidores de maior capacidade quer de uma maior largura de banda impõe custos adicionais que não são bem vistos pelos canais de distribuição. Este constrangimento reflecte-se na política vigente em alguns *sites* da Internet, onde é exigido o registo (e mesmo o pagamento) como via de acesso a conteúdos vídeo. Desta forma, corre-se o risco de um visionamento por uma franja restrita da ciberpopulação;
- Ainda relativamente à Internet, esta possui algumas limitações ao nível da implementação do Multimédia, sobretudo no que diz respeito à utilização de vídeo. As limitações de largura de banda lideram, de forma destacada, a lista dos problemas a considerar. Para citar um exemplo ilustrativo, caso se opte pelo *download* de um ficheiro de vídeo na Internet, o tempo de espera é elevado, variando consoante o tamanho do mesmo e a largura de banda disponível; caso a via *streaming* seja a opção, o vídeo será de qualidade inferior (susceptível a engasgos resultantes de flutuações de largura de banda), o tempo de arranque da aplicação de leitura do vídeo não é desprezível e, para terminar, poderá ser necessário responder de antemão a questões relativas a especificações técnicas, nem sempre compreendidas pelos utilizadores;
- O carácter democrático da Internet, onde qualquer pessoa tem a possibilidade de publicar os seus conteúdos num espaço de partilha a nível mundial, faz desta estrutura um mar imenso de pequenas notas pessoais, uma rede mundial baseada em "*hiperpost-it*". Face ao amadorismo reinante, os conteúdos de complexidade superior são muitas vezes desprezados, como é o caso do Vídeo;
- A falta de normalização é predominante, face às diversas aplicações de leitura de vídeo (*players*) nem sempre compatíveis entre si e, também, a múltiplos *codecs* utilizados. É comum a incapacidade de visualizar um vídeo, condicionado

pela não disponibilidade da aplicação ou o *codec* necessários para a sua leitura;

- Salvo raras excepções, a qualidade do Vídeo em qualquer produto Multimédia é inferior à que estamos habituados pelas tecnologias tradicionais, como é a televisão (quer por emissão terrestre, redes de televisão por cabo ou DVD Vídeo) e o cinema. Esta situação é motivada pelas limitações de largura de banda existentes, que levam a que o Vídeo veja algumas das suas variáveis comprometidas, como é o caso do número de imagens por segundo, a dimensão da imagem em *pixels* e a compressão empregue. Esta regressão qualitativa é mal compreendida pelos utilizadores. Chapman e Chapman [5] comparam o Vídeo em Multimédia a *«dancing postage stamps»*;
- As restrições técnicas e económicas dos utilizadores são uma barreira forte à generalização de sistemas baseados em videoconferência. São elas, por exemplo, a inexistência de altifalantes em alguns computadores, o tamanho restrito da moldura do Vídeo, o "roubar" espaço útil do ecrã a outras aplicações de importância superior, o incómodo que pode causar o som do Vídeo a colegas de trabalho fisicamente próximos, redes que não permitem o recurso a aplicações de *streaming*, computadores desactualizados não adequados ao fluxo do Vídeo, entre muita outras. A motivação para a compra de câmaras Vídeo para a *Web* (*webcameras*), que não estão incluídas no equipamento base de um computador pessoal (PC), não é grande, pois a sua utilidade é tanto maior quanto o número de utilizadores que as possuírem. Citando Egido, *«as with most interactive technologies, the more people that have videoconferencing, the more useful it becomes.»* [12];
- A importação indiscriminada das práticas vigentes em meios como o cinema e televisão para o Vídeo em Multimédia não é adequada. Para citar um exemplo, a gramática da edição praticada na televisão baseia-se na do cinema, tendo sido adaptada para dar resposta às suas particularidades, onde o tamanho do ecrã, o local de visualização e a predisposição do espectador são factores a considerar;
- Embora os Vídeos em Multimédia possam explorar outras proporções/formatos (altura *vs.* largura) que não o standard 4:3 ou 16:9 de forma a melhorar a integração em produtos Multimédia, é, no entanto, pouco comum encontrar Vídeos que não obedeçam a estes valores. Naturalmente que a tecnologia associada à produção de Vídeo limita as opções, mas não é factor impeditivo para a exploração de outros formatos;
- O Vídeo é normalmente associado a um veículo de lazer. Quando utilizado para outros fins, a resposta pode ser adversa, de desconfiança, não sendo encarado com a seriedade desejada;
- Segundo definem Fetterman e Gupta [13] e Götze, Boles *et al.* [14] podemos distinguir dois tipos de média: discretos e contínuos, também conhecidos como média estáticos e dinâmicos por Liestøl [15] e Branco [16]. A assunção de que estes dois tipos de média são totalmente compatíveis entre si poderá não ser totalmente verdadeira. Para Chambel [17], o actual conflito dominante no Multimédia reside na integração de meios espaciais e temporais, que se coloca ao nível tecnológico e cognitivo. Na óptica de Liestøl [15], vídeo e texto requerem diferentes abordagens por parte do utilizador. A leitura de texto requer a «combinação de letras que formam palavras, e palavras que formam frases»; o leitor decide o tempo e o ritmo da leitura. Com o Vídeo, o tempo é independente do espectador. Esta diferença fundamental prende-se com o conceito de linearidade em documentos hipermédia que alberguem diferentes média [15]. Guimarães, Chambel *et al.* [18] acrescentam que um problema fundamental entre o texto e o vídeo é a falta de ligação que existe entre estes dois tipos de informação e o formato monolítico do vídeo que leva a uma exploração menos efectiva desta fonte. No entanto, os autores defendem que a integração aumenta a eficácia através do reforço mútuo dos textos e do vídeo;
- Segundo Nardi, Schwarz *et al.* [19], a investigação que debate o valor do Vídeo baseia-se, na sua grande maioria, em vídeos com a abordagem *talking head* ("cabeça falante"), ou seja, vídeos nos quais nos é apresentado um ou mais oradores, em discurso directo para a câmara, ou em alguma actividade entre eles. Segundo os autores, esta abordagem tem conduzido a resultados negativos na apreciação da utilidade do Vídeo. Nardi, Schwarz *et al.* [19] defendem o *"video-as-data"* (Vídeo veículo de informação), em que o espaço de trabalho e os objectos de interesse estão no centro das atenções. Então, a exploração da telepresença[1] é relegada para segundo plano, estando a objectiva direccionada para outros objectos não necessariamente orgânicos. Neale, McGee *et al.* [22] afirmam que o *"video-as-data"* tem demonstrado *«the value of video for coordinating activities that primarily deal with physical artifacts remote from one or more of the coparticipants»*;
- A impossibilidade de transposição de um Vídeo para o papel, tal como se imprime um texto ou uma imagem, limita a omnipresença do Vídeo. A

[1] Este termo é, de acordo com Muhlbach, Bocker *et al.* [20] e Beigl e Gellersen [21], a sensação de estar a partilhar o mesmo espaço físico com pessoas que se encontrem localizadas remotamente. Nardi, Schwarz *et al.* [19] complementam a ideia, defendendo que a telepresença é a transferência de um contexto físico e psicológico para locais remotos.

visualização de um vídeo requer sempre a utilização de um dispositivo tecnológico, que é, para além de pouco prático, sempre mais caro;
- *«One of the first misconceptions about streaming video is that it is just an add-on. By that, I mean that many people think that you can take any video short and just 'add-on' streaming at the end of the process»* [23];
- A adopção de sistemas de videoconferência (e outras tecnologias Multimédia) tem sofrido de mitos de marketing que os promovem com substitutos de interacção presencial ("cara-a-cara"), gorando as expectativas dos utilizadores [12].

III. TAXONOMIA

Clarificados que ficaram os conceitos de Vídeo e Multimédia, é altura de proceder a uma taxonomia Vídeo com base no levantamento das soluções existentes, tanto a nível académico como comercial, e na bibliografia existente.

Numa primeira fase será caracterizada a integração de Vídeo em Multimédia e, posteriormente, a atenção centrar-se-á na classificação do Vídeo *per se*, nunca perdendo de vista o contexto do Multimédia.

A. Integração de Vídeo em Multimédia

Apesar da integração de Vídeo em Multimédia ser uma tarefa de dificuldade elevada, a sua taxonomia não obedece ao mesmo grau de complexidade. Atendendo ao panorama geral das aplicações, a inserção de Vídeo no Multimédia poderá dividir-se em três categorias gerais: Vídeo Independente, Vídeo Embebido e Vídeo Integrado (Cf. Figura 1).

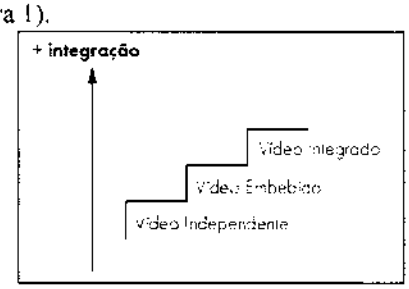


Fig. 1 - Categorias de Vídeo e integração Multimédia

O Vídeo Independente caracteriza-se por ser reproduzido num espaço independente do documento visualizado. Esta situação ocorre vulgarmente através da abertura de uma nova janela, onde o vídeo é reproduzido, seja esta janela a aplicação de leitura (Cf. Figura 2), ou uma janela convencional no mesmo "formato" da janela do documento principal.

Uma solução ligeiramente mais integrada (em contexto *Web)* é a proposta pelo *browser* Internet Explorer da Microsoft, que se processa da seguinte forma: caso o utilizador esteja interessado, este *browser* fracciona a janela em dois, reservando a moldura direita para o

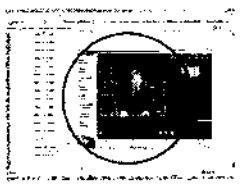


Fig. 2 – Vídeo Independente I

documento principal e a esquerda para o média a reproduzir (Cf. Figura 3).

Apesar da sua aplicação prática ser sofrível, consequência da diminuição do espaço útil para o documento principal, esta abordagem optimiza a ligação

Fig. 3 – Vídeo Independente II

entre o documento de origem e o média reproduzido. Desta forma, ambos estão inseridos numa única janela, de modo a resolver o problema de desagregação que a abertura de janelas múltiplas pode criar.

No patamar seguinte surge o Vídeo Embebido que se distingue do anterior pela inserção do vídeo no corpo do documento (Cf. Figura 4).

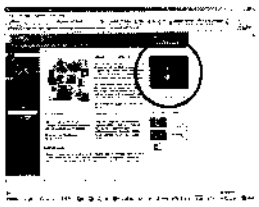


Fig. 4 – Vídeo Embebido

Apesar de se situar um degrau acima na escalada da integração, visível na Figura 1, a produção deste Vídeo não contempla preocupações com vista à sua inserção no documento (ou vice-versa). O vídeo é, como no Vídeo Independente, totalmente autónomo.

No último escalão surge o Vídeo Integrado, que não só possui o vídeo inserido no documento, característica por si suficiente para definir um Vídeo Embebido, como tem de possuir alguma integração com o documento envolvente. Tem de existir uma relação de interdependência entre os dois, fazendo com que a sua visualização fora do documento não permita a sua total compreensão.

A integração pode ocorrer a diversos níveis, desde um patamar mais básico, como a cor de fundo, até à comunicação de elementos do vídeo com a envolvente do documento. A Figura 5, retirada da página *Web* www.fullsail.com [24], apresenta um exemplo claro de integração, onde o vídeo (assinalado com o círculo) não tem margens definidas. Este efeito é conseguido através da utilização da mesma cor de fundo no vídeo e documento

(branco), combinação que cria a ilusão de imersão do indivíduo no ambiente gráfico.

Fig. 5 – Vídeo Integrado

Para além desta particularidade, o indivíduo interage com os componentes da interface gráfica, apontando para os elementos que respondem às suas acções, num jogo de acção-reacção.

B. Classificação de Vídeo

Após o desenho da taxonomia respeitante à integração de Vídeo em Multimédia, parte-se para a caracterização do Vídeo *per se*. A taxonomia basear-se-á em considerações formais da comunicação e, também, em aspectos tecnológicos. A sua estrutura assenta quase exclusivamente em sistemas conectados em rede por duas razões: primeiramente, porque estes sistemas contêm todas as particularidades de um vídeo convencional e, ao mesmo tempo, possuem algumas que lhe são exclusivas; em segundo lugar, porque o futuro do Multimédia (assim como já transparece no presente) estará indubitavelmente associado a sistemas de rede.

Após extensa consulta bibliográfica, Almeida [25], foi delineada a taxonomia apresentada na Figura 6.

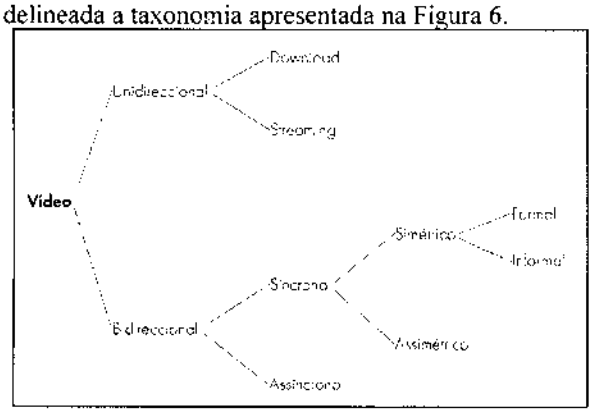


Fig. 6 - Taxonomia Vídeo (Almeida, 2004)

Antes de avançar será importante advertir que a bibliografia relativa a Vídeo e Multimédia teve o seu apogeu na década de 80 e inícios da década de 90, período em que se conjecturava que a utilização massiva de vídeo em sistemas Multimédia, sobretudo sistemas de vídeo bidireccional, seria uma realidade num curto espaço de tempo. Tal não sucedeu e a fase de euforia abrandou, dando lugar a uma fase também profícua, mas menos arrojada com algumas reservas. Deste modo, alguma bibliografia peca por desactualização e desajuste dos sistemas Multimédia que vigoram hoje em dia.

O Vídeo começa por ramificar-se em dois subgrupos: vídeo unidireccional e bidireccional. O primeiro – unidireccional – refere-se, como o próprio nome indica, a vídeo que decorre num único sentido. Ainda que possa existir interacção com o utilizador, como é o caso do vídeo interactivo, o canal vídeo não deixa de ser unidireccional. Este ramo subdivide-se ainda em duas categorias de natureza mais técnica: *download* e *streaming*. A primeira (*download*) refere-se a qualquer vídeo que esteja na sua totalidade no terminal onde vai ser visualizado; a segunda (*streaming*), e por oposição, aplica-se em situações em que o vídeo é visualizado à medida que é descarregado a partir de um terminal remoto.

Quanto ao segundo caso – vídeo bidireccional –, reporta-se a um sistema onde a transmissão vídeo ocorre nos dois sentidos, ou seja, se um terminal for receptor de vídeo será, também, emissor. Este tipo de solução é utilizado (quase) exclusivamente em situações de comunicação humano-humano, tentando o vídeo suprir a privação de contacto presencial. Esta categoria subdivide-se em síncrono e assíncrono, de forma a distinguir a comunicação que ocorre em tempo real, ao mesmo tempo (síncrono), daquela que ocorre em diferido, em espaços temporais diferentes (assíncrono).

A categoria síncrono ainda se subdivide em simétrico e assimétrico, permitindo diferenciar situações em que há diferença de fluxo de dados de vídeo. A utilização de um exemplo poderá ajudar a compreender este conceito: no caso de uma aula, em que para um professor temos diversos alunos, existe uma diferença de fluxo de dados entre as duas partes; no caso de comunicação um-para-um, a comunicação será potencialmente equitativa.

O ramo simétrico termina com uma distinção relativa à estrutura da comunicação: formal e informal. De modo breve, Fish, Kmut et al. [26] definem comunicação formal como aquela que circula através dos canais oficiais da organização, normalmente planeada e tipicamente conduzida num estilo "formal", "cerimonioso". Por oposição, a comunicação informal foge a estes canais oficiais e ocorre, frequentemente, de forma espontânea. Em termos de estilo, a comunicação informal acontece de forma mais frequente, mais expressiva, "descontraída" e interactiva.

IV. BOAS PRÁTICAS E PARADIGMAS NA PRODUÇÃO E INTEGRAÇÃO DE VÍDEO PARA MULTIMÉDIA

A integração de Vídeo na esfera do Multimédia, sector onde a limitação de largura de banda é uma constante, veio obrigar à adaptação e redefinição de algumas das boas práticas e paradigmas vigentes neste média. De facto, tal como aconteceu com o cinema e a televisão, o Vídeo necessitou, na transição para este meio, da reestruturação de uma linguagem que, até então, era detentora de alguma estabilidade. Com esta ideia em mente, é objectivo da presente secção pesquisar e compilar boas práticas na produção de vídeo, com vista à integração em sistemas Multimédia unidireccionais ou bidireccionais.

O ciclo de produção é, no audiovisual, geralmente dividido nas fases de pré-produção, produção e

pós-produção. No entanto, nos manuais que tratam a relação entre Vídeo e Multimédia, esta estrutura não é habitualmente aplicada na análise das suas componentes. De facto, existe uma série de tarefas particulares que levam à reestruturação deste processo, sobretudo na fase de pós-produção.

Com base na decomposição do ciclo de produção definidos em [27] e [28], optou-se por dividir a presente secção do seguinte modo: Criação de Conteúdos, Transferência/Digitalização, Edição, Integração, Distribuição e Visualização, apresentadas de seguida:

Criação de Conteúdos

- O relatório do Adobe Dynamic Media Group [27] aconselha a que se filme em exclusividade para *streaming*. Em concordância, Lopes, Moreira *et al.* [6] afirmam: «Na maior parte dos casos e aplicações, fazem-se transcrições de conteúdos que na realidade são originalmente provenientes da indústria do cinema ou vídeo, recodificados num formato digital para ser incluído em Multimédia, conteúdos que não foram pensados, de origem, para o produto multimédia final»;
- A opção por uma abordagem minimalista é aconselhada, definida como uma técnica artística caracterizada pela extrema frugalidade e simplicidade. «*In other words, keep it simple!*» [27]. Conforme afirmam Kelsey e Feeley [29], «*Blair Witch, MTV, and even VH1 "looks" are out*». A complexidade e movimento são factores a evitar, sem no entanto entediar os espectadores com conteúdos demasiadamente estáticos: «*Action! (just don't move around very much)*» [27];
- Kelsey e Feeley [29] aconselham, no mínimo, a filmar com recurso a câmaras DV, considerando também a utilização dos formatos superiores DVCAM, DVCPRO, Digital-S. Segundo os autores, os últimos formatos dão acesso a câmaras de boa qualidade dotadas de boas objectivas, com possibilidades acrescidas no controlo manual dos parâmetros, resultando em vídeo de qualidade superior. Todavia, em [27] é afirmado que «*streaming media can be "forgiving"*», ou seja, apesar da opção pelo melhor equipamento ser desejável, o *streaming* de vídeo é mais complacente com a baixa qualidade do que outros formatos de qualidade superior;
- A utilização de tripé é crucial [23, 27, 28, 30] como forma de eliminar movimentos desnecessários da câmara que podem resultar em dificuldades de compressão. Como afirmam Thornhill, Asensio *et al.* [28], técnicas de filmagem sem suporte (*handheld*) não funcionam. Caso não seja possível utilizar um tripé, a opção pode incidir sobre um monopé [23]. Se, ainda assim, nenhuma destas alternativas for viável, é importante explorar formas de fixar a câmara, como encostar o operador de câmara a uma parede [23], apoiar os braços nas costas de uma cadeira [23, 31], entre outras;
- No seguimento do ponto anterior, a utilização de panorâmicas (horizontais e verticais) e a variação da distância focal enquanto se filma (*zoom in* e *zoom out*) devem ser evitadas [27, 28, 30]. Se a utilização de panorâmicas for impossível, é conveniente a utilização de um tripé de qualidade para reduzir o movimento a uma única dimensão [27];
- O fundo deve ser o mais simples possível [27, 28, 30, 32]. Deste modo, texturas e padrões complexos devem ser evitados por provocarem efeitos de distorção [27]. Folhas de árvores ao vento são proibidas [32]. Duas estratégias possíveis para tornar o fundo mais simples e homogéneo são técnicas de *bluescreen* [27] (aplicando um fundo estático e de complexidade reduzida sob a figura), ou desfocando o fundo através da redução da profundidade de campo [28, 30];
- Quanto à composição da imagem, e para além dos referidos padrões complexos a evitar (como riscas, xadrez, entre outros), também as cores saturadas podem causar problemas [27, 30]. Tons escuros e tons terra obterão melhores resultados [27]. A cara dos participantes deve ser a zona de maior luminância da imagem [33]. A utilização de fundos brancos é desaconselhada, pelo que é essencial ter em atenção *whiteboards* e projecções como pano de fundo. As cores do espaço envolvente devem situar-se em tons médios e, de preferência, com acabamento mate [34];
- Ao filmar pessoas, Kennedy [23] tem uma regra: aproximar do sujeito, até ao ponto em que este parece psicologicamente demasiadamente próximo; nessa altura, aproxima-se ainda mais. O autor diz que existe uma série de razões para seguir esta prática, relembrando que o vídeo será exibido num tamanho muito inferior ao de *full screen*. Os grandes planos resultam melhor, permitindo aos "Webespectadores" reconhecer faces e expressões, ou distinguir objectos e seus detalhes [27]. A este respeito, Thornhill, Asensio *et al.* [28] afirmam que o plano se deve concentrar no apresentador e cortar o mais possível de fundo, estando preparado para movimentos súbitos do sujeito que o podem retirar do plano;
- No caso de uma comunicação, o orador deve manter contacto visual com a câmara, condição difícil de obter quando na presença simultânea de uma plateia. No entanto, a percepção de que o público via *Web* é, também, parte activa da comunicação, ajudará a captar a atenção dos participantes remotos. Alguns oradores filmam previamente um plano dirigidos para a câmara para saudar estes participantes [28];

- Após investigação respeitante ao grau de precisão com que um indivíduo consegue aferir se um "olhador"[2] está efectivamente a estabelecer contacto visual (*eye contact*), Chen [35] retira duas conclusões: primeiramente, existe uma área na qual, ainda que o "olhador" não olhe directamente nos olhos do indivíduo, este irá interpretar como contacto visual; em segundo lugar, esta sensibilidade é assimétrica, ou seja, o indivíduo é menos sensível nesta aferição quando o "olhador" olha para baixo, quando comparado com a sensibilidade nas outras direcções. Neste sentido, de forma a optimizar o contacto visual em situações de vídeo bidireccional, a câmara deve encontrar-se o mais próximo possível da imagem do participante remoto e, igualmente, acima da imagem;
- O canal áudio não deve, de forma alguma, ser descurado. Fox [36] afirma que em filmagens de orçamento limitado é um erro dispensar um técnico de som. De acordo com [37], o melhor do vídeo na *Web* não é a imagem mas o som: enquanto que a qualidade da imagem é mínima, é possível atingir áudio de grande qualidade. «*Here is where you want to shine*» [37]. O relatório "*A Streaming Media Primer*" [27] é da mesma opinião, defendendo que os utilizadores podem perdoar mau vídeo, mas abandonarão a visualização por mau áudio;
- No que respeita ao equipamento de captação de áudio, uma regra é consensual: evitar a utilização do microfone incorporado da câmara [27-29, 37];
- Na sua forma presente, o *streaming* não é um meio marcado pela subtileza. Se existir narração, os espectadores têm de a ouvir de forma clara, pelo que [38] aconselha a "*bury that background music deep*", ou seja, favorecer o diálogo em detrimento dos outros elementos sonoros. A música ajuda a estabelecer o ambiente, mas é a voz que vende, informa ou educa [38];
- Em [27] é recomendada a utilização de som mono, uma vez que, em situações de *streaming* de vídeo, o som estéreo será um desperdício de largura de banda;
- De acordo com Beggs e Thede [39], o processo de compressão áudio acentua ruídos de fundo que não são percebidos na fonte sonora original. Como tal, e sabendo que as frequências acima de 10KHz e abaixo dos 75 Hz transportam ruído, a utilização de filtros passa alto e passa baixo poderão corrigir este problema [39];
- [39] recomendam a normalização do áudio a 95%. Esta operação, que consiste em ajustar a amplitude da onda até que a amostra mais alta se aproxime do topo da escala (sem exceder), permite uniformizar o volume entre múltiplas fontes sonoras, maximizando a gama dinâmica [39].

Transferência/Digitalização

No capítulo da transferência ou digitalização do Vídeo para o terminal responsável pela edição, não há nenhuma prática passível de referência. «*If possible, keep it digital*» [27], de forma a evitar perdas em processos de conversão entre analógico e digital.

Edição

- Tornar o vídeo progressivo (não entrelaçado). Genericamente, o vídeo em computador pessoal, nomeadamente *streaming*, é beneficiado neste formato [28];
- Citando Moutain, Fox [36] sugere que se inicie o vídeo com um *fade* de preto para a imagem. Caso contrário, ocorrerá a passagem de «*zero to having a lot of information*». Um *fade* irá moderar a quantidade de nova informação por unidade de tempo e evitar a *pixelização*. Ainda neste contexto, Fox [36] refere também Brown, que afirma que mistura e cortes simples são normalmente as melhores transições a aplicar, ideia partilhada por Thornhill, Asensio *et al.* [28]. Estes autores afirmam que os sistemas de edição oferecem muitas transições vídeo e áudio que, para além de estética duvidosa, são um trabalho extra para o compressor [28];
- Quando é necessária a titulagem do vídeo, a reduzida dimensão do ecrã leva a que, por motivos de legibilidade, os títulos ocupem uma área superior da imagem, quando comparados com a televisão [27, 29, 40]. Deste modo, é importante utilizar tipos de letra com dimensão superior. Fox [36] sugere a adição de uma barra preta (ou de uma cor que contraste) sob os títulos ou, em alternativa, colocar a titulagem em áreas da imagem com pouco movimento e pouco descritivas. Uma vez que pode demorar algum tempo até que o *software* acompanhe todas as mudanças na imagem, a titulagem deve permanecer no ecrã mais tempo que o habitual;
- Fox [36] afirma que vale a pena cortar as primeiras e últimas linhas da imagem que geralmente contêm ruído. Nesta operação deve ter-se o cuidado de assegurar que a dimensão em *pixels* do vídeo seja divisível por quatro para ser matematicamente compatível com a maioria dos *codecs* [41]. Thornhill, Asensio *et al.* [28] sugerem mesmo que se cortem espaços não úteis da imagem. Na verdade, motivado pela diminuição do tamanho da imagem original, é possível reenquadrar um plano. Assumindo que o original tem a dimensão de 720 por 576 *pixels* e o vídeo final 176 por 144, é possível proceder a esta operação sem perda de

[2] Apesar do termo "olhador" ser demasiado informal, revelou-se a melhor tradução para a palavra inglesa *looker*.

qualidade. Assim, será possível transitar de um plano geral para um plano mais próximo, já que o Vídeo para o Multimédia assenta sobre grandes planos;
- Os vídeos deverão ser pequenos em duração [28, 42] e segmentados em capítulos [42], determinado em parte pela pouca atenção que os espectadores dedicam a olhar para um ecrã de computador [28]. Enquanto que Thornhill, Asensio *et al.* [28] afirmam que não existe nenhuma duração recomendada, variando consoante o tema do vídeo a exibir, Nielsen [42] recomenda o máximo de 1 minuto.

Integração

- Motivado pela reduzida qualidade do *streaming* é preferível fornecer uma versão de qualidade superior via *download* (note-se que esta recomendação é do ano de 1999, pelo que pode pecar por falta de actualidade) [42];
- Para qualquer *download* que demore mais de 10 segundos é conveniente fazer referência ao tamanho do ficheiro e à duração do vídeo [42];
- A exibição de uma ou duas imagens estáticas retiradas do vídeo, e/ou um pequeno resumo escrito do conteúdo devem ser apresentados. Só o título não é, com frequência, suficiente para a percepção do seu conteúdo [42];
- Liestøl [15] cita Kahn e Haan [43] onde é discutido o problema da criação de *links* em pontos de decisão. Os autores concluem que, idealmente, estes devem surgir como sinais visuais na superfície do vídeo, face à sua relevância ao longo do tempo. Esta afirmação é sustentada pelo facto dos *links* necessitarem de estar estreitamente relacionados com material com que estabelecem ligação e, também, a existência de *links* deve ser exibida sem dispersar a atenção do utilizador da fonte;
- Relativamente à compressão, este é um processo com poucas funções parametrizáveis, no qual o utilizador tem pouca intervenção, ou, pelo contrário, reveste-se de uma complexidade extrema, cuja abordagem não se adequa ao presente trabalho. As únicas considerações que podem aqui ser apontadas são as referidas no relatório do Adobe Dynamic Media Group [27], que aconselha a que, no caso de uma distribuição em Intranet, se seleccione um único formato com a *bit rate* optimizada, ou, no caso de distribuição em Internet, se facilitem múltiplos formatos a diversas *bit rates* [27]. Ainda em Internet, Fox [36] sugere a distribuição em duas versões (alta e baixa largura de banda) e dois formatos, sendo um o QuickTime e o outro à escolha entre o Windows Media Player e o Real. Beggs e Thede [39] sugerem que em situações onde o elevado grau de movimento associado a reduzida largura de banda acarrete imagens "*blurry*" e indistintas, é aconselhável optar por diminuir o número de imagens por segundo para 1, resultando num *slideshow* com imagens de qualidade superior.

Distribuição, Visualização

A questão das boas práticas nas secções de Distribuição e Visualização foge já ao âmbito das recomendações estabelecidas nas secções anteriores. Deste modo, e dada a extensão das mesmas, optou-se por não as incluir neste artigo.

V. CONCLUSÕES

As dificuldades de integração de Vídeo em Multimédia são inegáveis. Como consequência, a utilização deste média tem, efectivamente, sido diminuta, ainda que considerado como um dos pilares fundamentais da matriz do Multimédia e, cumulativamente, detentor de um extenso espectro de vantagens.

No capítulo das boas práticas, as sugestões descritas são, maioritariamente, de ordem muito técnica, indiciando uma lacuna na preocupação com questões de outro teor. No entanto, o seu cumprimento poderá revelar-se uma mais valia na criação de conteúdos Vídeo para Multimédia.

Em suma, o processo de produção de Vídeo para Multimédia reveste-se de inúmeras especificidades que têm de ser tidas em consideração logo na fase de planeamento. A enumeração de boas práticas apresentada –que estará ainda longe de todos os aspectos a contemplar– confirma que a criação de Vídeo para Multimédia necessita de um espaço e uma produção próprias, não passando pela compressão de um vídeo concebido para outro suporte. Esta abordagem traduz-se, necessariamente, em necessidade de capital para uma produção integrada de Vídeo para Multimédia.

REFERENCES

[1] N. Dimitrova, R. Koenen, H. Yu, A. Zakhor, F. Galliano, e C. Bouman, "Video portals for the next century (panel session)", 7th ACM international conference on Multimedia, Orlando, 1999.

[2] P. Lopes, M. Moreira, e N. Santos, "Vídeo Digital para Multimédia: Boas Práticas de Aquisição e Processamento", 12º Encontro Português de Computação Gráfica, Porto, 2003.

[3] F. Simões e M. Pinto, Perspectivas de Futuro: Som Imagem Interactividade e Multimédia, vol. 12. Rio Tinto: Edições Asa, 1995.

[4] F. Fluckiger, Understanding networked multimedia: applications and technology. Hertfordshire: Prentice Hall, 1995.

[5] N. Chapman e J. Chapman, Digital Multimedia. Chichester, UK: Wiley, 2002.

[6] P. Lopes, M. Moreira, e H. Pereira, "Estratégias de Desenvolvimento de Jogos Multimédia", 10º Encontro Português de Computação Gráfica, Lisboa, 2001.

[7] P. Lopes, "Multimédia: rápido, divertido, fácil e barato", Seminário de Multimédia e Computação Gráfica XXI - O Futuro, organização do GPCG e APDC, Multimédia XXI, Feira Internacional de Lisboa, Lisboa, 2001.

[8] B. Bailey, J. Konstan, e J. Carlis, "Supporting Multimedia Designers: Towards More Effective Design Tools", Multimedia Modeling, 2001.

[9] L. Marks e B. Davis, "Integrative Multimedia Design", Conference Companion on Human Factors in Computing Systems, Boston, 1994.

[10] P. Lopes, "conversa pessoal", Lisboa, 2003.

[11] M. Gleicher, Rachell Ieck, e M. Wallick, "A framework for virtual videography", 2nd International Symposium on Smart Graphics, Hawthorne, 2002.

[12] C. Egido, "Video conferencing as a technology to support group work: a review of its failures", 1988 ACM Conference on Computer-Supported Cooperative Work, Portland, 1988.

[13] R. Fetterman e S. Gupta, Mainstream Multimedia: Applying Multimedia in Business. Londres, 1993.

[14] R. Götze, D. Boles, e H. Eirund, "Multimedia User Interfaces", University of Oldenburg, Oldenburg 1996.

[15] G. Liestol, "Aesthetic and rhetorical aspects of linking video in hypermedia", 1994 ACM European Conference on Hypermedia Technology, Edinburgh, 1994.

[16] V. Branco, "Notas da disciplina de Ferramentas Multimédia, Mestrado em Gestão da Informação", Universidade de Aveiro, 2003.

[17] T. Chambel, "Integração e Sincronização Multimédia na Web", Lisboa, 2004.

[18] N. Guimarães, T. Chambel, e J. Bidarra, "From Cognitive Maps to Hypervideo: Supporting Flexible and Rich Learner-Centred Environments", Interactive Multimedia Electronic Journal of Computer-Enhanced Learning, Vol. 2, 2000.

[19] B. Nardi, H. Schwarz, A. Kuchinsky, e R. Leichner, "Turning Away from Talking Heads: The Use of Video-as-Data in Neurosurgery", SIGCHI Conference on Human Factors in Computing Systems, Amesterdão, 1993.

[20] L. Muhlbach, M. Bocker, e A. Prussog, "Telepresence in videocommunications: A study on stereoscopy and individual eye contact", Human Factors, 1995.

[21] M. Beigl e H.-W. Gellersen, "Ambient Telepresence", Workshop on Changing Places, Londres, 1999.

[22] D. Neale, M. McGee, B. Amento, e P. Brooks, "Making Media Spaces Useful: Video Support And Telepresence", Virginia Polytechnic Institute and State University, Blacksburg 1998.

[23] T. Kennedy, "Streaming Basics: Shooting Video for Streaming", http://smw.internet.com/video/tutor/streambasics1, consultado em 2004-01-27, 2001.

[24] Fullsail, http://www.fullsail.com, consultado em 2004-01-12, 2003.

[25] A. Almeida, "Sistemas de Comunicação e Multimédia com Integração de Video: evolução, situação actual e boas práticas" Dissertação de Mestrado, Universidade de Aveiro, 2004.

[26] R. Fish, R. Kmut, R. Root, e R. Rice, "Video as a technology for informal communication", Commun. ACM, vol. 36, pp. 48-61, 1993.

[27] Adobe Dynamic Media Group, "A Streaming Media Primer", 2001.

[28] S. Thornhill, M. Asensio, e C. Young, "Video Streaming: a guide for educational development", Manchester 2002.

[29] L. Kelsey e J. Feeley, "Shooting Video for the Web", in DV, 2000, pp. 54-62.

[30] Integrated Streaming, "Tips: Video for the Web", http://www.integratedstreaming.com, consultado em 2004-05-18.

[31] J. Edgecoe, Guia Completo do Vídeo. Lisboa: Dinalivro, 1997.

[32] L. Kelsey e J. Feeley, "Shooting Video for the Web", http://www.mssvision.com, consultado em 2003-10-12.

[33] J. Goldstein e P. Bagdon, "Success Without Boundaries - Wealth Without Risks", http://picturephone.com/book_download_approved.htm, consultado em 2004-06-10.

[34] ViDe, "Videoconferencing Cookbook", http://www.videnet.gatech. edu/cookbook, consultado em 2004-02-19.

[35] M. Chen, "Leveraging the asymmetric sensitivity of eye contact for videoconference", SIGCHI Conference on Human Factors in Computing Systems, Minneapolis, Minnesota, 2002.

[36] C. Fox, "Shooting for the Web", http://desktopvideo.about.com, consultado em 2004-05-18.

[37] University of Wisconsin Eau Claire, "Tips for Shooting Video for Web Streaming", http://www.uwec.edu, consultado em 2003-10-09.

[38] T. Kennedy, "Streaming Basics: Editing Video for Streaming", http://smw.internet.com/video/, consultado em 2004-05-18, 2001.

[39] J. Beggs e D. Thede, Designing Web Audio. Sebastopol: O'Reilly & Associates, 2001.

[40] S. Schenk, "Web Tips: Shooting Video for the Web", http://www.creativepro.com, consultado em 2003-10-09, 2003.

[41] Broadbandlab, "Editing video for streaming", http://www.broadbandlab.org.uk/19.0.html, consultado em 2004-06-05.

[42] J. Nielsen, "Video and Streaming Media", http://www.useit.com, consultado em 2002-08-20, 1999.

[43] P. Kahn e B. Haan, "Video in Hypermedia: The Design of InterVideo" Visual Resources, Vol. VII, pp. 353-360, 1991.

Uma Disciplina Introdutória à Interacção Humano-Computador: Aulas Práticas

Beatriz Sousa Santos

Resumo– **Tanto a tecnologia como a Interacção Humano-Computador evoluem rapidamente, pelo que se torna necessário um esforço contínuo no sentido de manter um conjunto de trabalhos práticos numa disciplina introdutória a esta área científica, que não só ajude os alunos a consolidar os conhecimentos adquiridos nas aulas teóricas, mas que seja também consentâneo com a tecnologia actual. Neste artigo descrevem-se resumidamente os temas abordados e os trabalhos realizados nas aulas práticas da disciplina Interfaces Humano-Computador, opção da Licenciatura em Engenharia Electrónica e de Telecomunicações da Universidade de Aveiro.**

Abstract– **Both technology and Human-Computer Interaction are evolving rapidly, thus a continuous effort is needed to maintain a set of practical assignments of an introductory course on that scientific area that not only helps students to consolidate the concepts acquired along the course, but also is adapted to the current technology. This paper briefly presents the practical classes and assignments of an introductory course on Human-Computer Interfaces offered as elective to Electronics and Telecommunications Engineering students at the University of Aveiro.**

I. INTRODUÇÃO

Actualmente os utilizadores de sistemas computacionais interactivos frequentemente não possuem literacia computacional, pelo que aqueles sistemas têm que ser desenvolvidos tendo em consideração as necessidades, capacidades e limitações dos utilizadores alvo, sendo desejável a utilização de um ciclo de desenvolvimento de *software* interactivo centrado no utilizador, tornando-se o problema do desenvolvimento da interface de utilizador ainda mais importante [1].

Por outro lado, como em qualquer outra disciplina de engenharia, a optimização de apenas uma parte de um sistema pode tornar inválido um projecto, que em princípio, tem uma contexto mais abrangente. Sendo assim, mesmo de um ponto de vista estritamente técnico é vantajoso formular o problema da interacção humano-computador de forma suficientemente ampla, ajudando os alunos e futuros profissionais a evitar um projecto divorciado do contexto do problema [2]. Acresce ainda que a interface de utilizador de um sistema computacional interactivo representa tipicamente mais de metade das linhas de código [3], sendo portanto necessário decidir cuidadosamente qual a funcionalidade a incluir no sistema, como fornecê-la ao utilizador e como testá-la.

Tendo em consideração as razões acima expostas, bem como o facto de vários autores [4-6], e os relatórios sobre os *curricula* nas áreas das Ciências e Engenharia de Computadores [7-8] defenderem que aqueles *curricula* devem reflectir a crescente importância da interface com o utilizador, foi criada em 1993/94 na Universidade de Aveiro uma disciplina introdutória a esta área científica. Esta disciplina tem sido oferecida como disciplina de opção no 5º ano da Licenciatura em Engenharia Electrónica e de Telecomunicações (LEET), tendo sido durante alguns anos oferecida também ao Mestrado em Engenharia Electrónica e de Telecomunicações. Mais recentemente tem sido oferecida como disciplina obrigatória à licenciatura em Engenharia de Computadores e Telemática, uma disciplina com uma carga horária e um conteúdo programático diferentes [9], embora na mesma área científica.

Trata-se de uma disciplina introdutória, pretendendo-se expor os alunos aos conceitos básicos da área. Os objectivos específicos mais importantes são:

1- Sublinhar a importância de um bom projecto da interface de utilizador;
2- Introduzir ferramentas, técnicas e ideias para o projecto de interfaces de utilizador;
3- Tornar mais fácil a comunicação entre os alunos (futuros engenheiros) e os especialistas em interacção humano-computador.

Pretende-se também, nesta disciplina, promover capacidades importantes como o raciocínio crítico, trabalho em grupo e comunicação oral e escrita.

Neste artigo apresenta-se resumidamente o conteúdo programático da disciplina e descrevem-se com mais algum detalhe os temas abordados nas aulas práticas e os trabalhos nelas realizados.

II. PROGRAMA

Como anteriormente referido, a Interacção Humano-Computador é uma área científica que evolui muito rapidamente, pelo que uma disciplina nesta área tem que ser cuidadosamente planeada por forma que os conceitos nela abordados não fiquem rapidamente desactualizados. Isto é particularmente importante já que, embora o

conteúdo programático de uma disciplina possa ser actualizado, em geral não é fácil aos engenheiros voltarem à Universidade para fazerem alguma actualização, devendo a maioria evoluir com base nos fundamentos obtidos enquanto eram estudantes.

De acordo com o relatório *Curricula for the Human-Computer Interaction* elaborado pelo ACM-SIGCHI [2]: "Human-Computer Interaction is a discipline concerned with the design, evaluation and implementation of interactive computing systems for human use and the study of major problems surrounding them". Sendo assim, envolve aspectos de ciência, engenharia e *design*, sendo abordados ao longo da disciplina os seguintes temas: princípios e paradigmas de usabilidade; perfil do utilizador e modelos mentais e conceptuais, dispositivos de entrada e saída; modelos a usar no projecto; estilos de diálogo; disposição de informação no ecrã e utilização da cor; tempo de resposta; documentação para utilizador; avaliação de interfaces de utilizador; *software* para interfaces de utilizador. Uma descrição mais detalhada do programa teórico da disciplina encontra-se em [9-10]

Como bibliografia são usados vários livros, *sites* e artigos, sendo as principais referências os livros [11-13].

III AULAS PRÁTICAS

Os trabalhos práticos realizados no âmbito desta disciplina têm variado ao longo dos anos, como conseqência de variações da carga horária da disciplina, do número e preparação dos alunos, bem como da evolução na área e mudanças na tecnologia.

Por motivos de ordem pedagógica e de motivação, é conveniente que alguns dos trabalhos práticos correspondam a necessidades reais, no âmbito de trabalhos de desenvolvimento ou de investigação. Sendo assim, sempre que possível, a autora inclui um trabalho que corresponde a projectar e implementar um protótipo de uma interface de utilizador para uma aplicação ou a participação em testes de usabilidade ou experiências controladas como utilizadores ou avaliadores para um "cliente" exterior à disciplina. Estes clientes têm sido sobretudo estudantes de pós-graduação e alguns dos trabalhos desenvolvidos neste âmbito têm sido apresentados em publicações nacionais, mas também em publicações internacionais; um exemplo de uma interface de utilizador desenvolvida numa das edições da disciplina encontra-se descrita em [14].

Durante os primeiros anos em que foi leccionada, a disciplina não tinha aulas práticas, era oferecida simultaneamente a alunos do 5º ano da LEET e do Mestrado, sendo frequentada por um número reduzido de alunos (entre doze e dezasseis). Nesta fase, foram realizados trabalhos envolvendo seis ou oito alunos de licenciatura coordenados por um aluno de Mestrado que tinha a responsabilidade de integrar o trabalho produzido pelos outros alunos. No sentido de orientar estes trabalhos eram realizadas reuniões com todos os alunos envolvidos e, frequentemente, com um especialista do domínio de aplicação. A autora considera que esta experiência foi geralmente muito positiva e que esta abordagem é particularmente interessante já que a necessidade de uma boa estratégia de trabalho em grupo se torna mais evidente à medida que a dimensão das equipas aumenta; no entanto, exige um controlo apertado do trabalho de cada membro do grupo, caso contrário a probabilidade de se obter uma colecção de módulos que não se conseguem integrar é elevada.

Nesta primeira fase da disciplina alguns alunos de Mestrado realizaram trabalhos sob temas propostos por eles próprios, sendo no entanto esses temas sujeitos à aprovação da professora. Estes trabalhos envolviam o projecto de uma interface de utilizador, a implementação de um protótipo e a sua avaliação. Estas interfaces destinavam-se em regra a ser utilizadas noutro contexto. Alguns destes trabalhos foram apresentados em publicações internas ou conferências nacionais.

Mais tarde, a disciplina passou a ser oferecida apenas como opção do 5º ano da LEET e começou a ter aulas práticas (duas horas semanais). O número de alunos aumentou para uma média de 20 alunos e os trabalhos práticos passaram a ser mais guiados. Esta alteração no tipo de trabalhos práticos ficou-se a dever não só ao aumento do número de alunos, mas também a uma recomendação no sentido de evitar a prática generalizada de realizar a avaliação prática através de mini-projectos, o que estava a ter um impacto negativo na dedicação dos alunos ao projecto de licenciatura.

Nesta segunda fase da disciplina segue-se a sugestão dada em [15] quanto à organização dos trabalhos práticos, sendo os primeiros trabalhos práticos dedicados à avaliação de interfaces de utilizador e o último trabalho prático ao *design* e implementação de um protótipo.

Além destes trabalhos práticos, tem-se dedicado algum tempo à introdução de duas ferramentas que permitem a prototipagem rápida de interfaces de utilizador: o Visual Basic e o HTML. Depois desta introdução, que consiste na descrição das características fundamentais da linguagem, os alunos são aconselhados a realizar alguns exemplos. A autora acredita que esta introdução, embora breve, é importante para motivar os alunos a aprofundarem a utilização destas linguagens que lhes poderão vir a ser úteis na sua vida profissional.

A tabela 1 mostra os temas que têm sido abordados nos trabalhos práticos (realizados em grupos de dois alunos) nos últimos anos, bem como o número de horas dedicado a cada um deles durante as aulas práticas. Nas secções seguintes dão-se mais alguns detalhes.

A. Trabalhos práticos

O facto de se abordar em primeiro lugar o estudo dos métodos de avaliação de interfaces de utilizador tem a grande vantagem de permitir aos alunos sentir as dificuldades dos utilizadores e de os motivar a estudar outros assuntos abordados na disciplina. Na realidade, de acordo com J. Nielsen [16], a maioria dos programadores, profissionais ou estudantes, mudam de atitude depois de presenciarem testes em que utilizadores "lutam" com

aplicações supostamente fáceis de utilizar. Ainda de acordo com o mesmo autor, este efeito é ainda maior se o *software* em teste tiver sido desenvolvido pelos próprios. Sendo assim, para promover esta mudança de atitude, as primeiras aulas práticas são dedicadas à aplicação de métodos de avaliação como as técnicas de observação, experiências controladas e avaliação heurística, em que os alunos actuam não só como avaliadores mas também como utilizadores. Nestas aulas avaliam-se sobretudo aplicações, *sites* da Web e telemóveis, no entanto também têm sido avaliados dispositivos interactivos como fotocopiadoras, câmaras digitais, gravadores video e outros. O trabalho final inclui também alguma avaliação.

Tabela 1- Temas abordados nas aulas práticas

Tipo	Tema	Horas de aula
Avaliação	Técnicas de observação	2
Avaliação	Experiências controladas	2
Avaliação	Avaliação heurística	4
Avaliação	Avaliação de telemóveis	2
S/W	Introdução ao Visual Basic	2
S/W	Introdução ao HTML	2
Design	*Design* de uma interface de utilizador	10

Trabalhos sobre avaliação de interfaces

O primeiro trabalho prático versa a aplicação de métodos de observação para a avaliação de *sites* de comércio electrónico, sendo definidas tarefas representativas neste contexto. Um aluno de cada grupo tem que desempenhar estas tarefas num dado *site*, enquanto o colega o observa e regista um conjunto de medidas de usabilidade comuns. Finalizadas estas tarefas, os alunos trocam de papel e repetem o trabalho mas noutro *site*. No fim, é preenchida uma tabela para cada *site*, sendo estes comparados no sentido de se extraírem algumas conclusões sobre a sua usabilidade. Têm sido usadas, neste trabalho, duas livrarias *on-line* (diferentes, ou a mesma mas em países diferentes), o que permite observar que distintos *designs* (por vezes apenas com diferenças subtis), embora ofereçam basicamente a mesma funcionalidade e disponibilizem a mesma informação, apoiam o utilizador de forma bastante diferente em tarefas específicas. Este trabalho tem também a vantagem de permitir aos alunos familiarizarem-se com as medidas de usabilidade mais comuns e constatarem as dificuldades que utilizadores, mesmo possuindo uma literacia informática em geral elevada, experimentam ao utilizar interfaces de utilizador que não conhecem. Uma vez que este é o primeiro trabalho prático, é fornecido um enunciado que descreve de forma bastante pormenorizada as tarefas a executar, as medidas a usar, as perguntas a fazer ao utilizador e como observar o utilizador.

No segundo trabalho prático, que trata o tema experiências controladas, pede-se aos alunos que participem como utilizadores numa experiência controlada. Em geral corresponde a uma experiência real, levada a cabo no âmbito do trabalho de investigação da autora e dos seus alunos de pós-graduação. Os alunos são livres de não participarem, mas têm que permanecer na aula; no entanto a grande maioria participa empenhadamente. O procedimento adoptado é exactamente o usado noutras sessões da mesma experiência conduzidas com outro tipo de utilizadores. No fim da sessão são explicados os principais aspectos da experiência, incluindo a motivação e objectivo, as hipóteses formuladas, as variáveis dependentes e independentes e o método experimental usado. Nos casos em que se conseguem obter alguns resultados ainda durante o semestre, estes são apresentados numa outra aula prática.

O terceiro trabalho prático consiste na avaliação heurística de uma aplicação, *site* ou dispositivo interactivo. Os alunos podem escolher o que vão avaliar, sendo, no entanto, esta escolha sujeita a validação pela autora. No âmbito deste trabalho, cada aluno realiza uma apreciação global do objecto da avaliação, no sentido de compreender quais são os utilizadores alvo, qual a funcionalidade e informação oferecidas, bem como qual a estrutura geral em que estas são disponibilizadas ao utilizador. Depois procede a uma análise mais detalhada, tendo por base as dez heurísticas de usabilidade propostas por J. Nielsen [17], devendo resultar numa lista de problemas de usabilidade com a respectiva classificação de gravidade de acordo com uma escala também proposta pelo mesmo autor. Na semana seguinte, os dois elementos de cada grupo confrontam as suas análises e elaboram um relatório conjunto.

O quarto trabalho prático consiste na proposta de um teste de usabilidade destinado a avaliar interfaces de utilizador de telemóveis. Para a realização deste trabalho os alunos devem tomar como base o *Common Industry Format Usability Test Reporting* (que se tornou na norma ANSI-354 em 2001) [18]. Cada grupo deve propor o tipo de utilizadores, as tarefas que estes devem executar e as medidas de usabilidade a usar; depois é efectuada uma discussão generalizada na aula, escrevendo-se no quadro uma proposta conjunta. Em seguida, cada grupo executa uma parte do teste utilizando um modelo de telemóvel desconhecido de, pelo menos, um dos elementos do grupo que funcionará como utilizador, enquanto o outro elemento regista as medidas de usabilidade. Finalmente, cada grupo reporta informalmente a toda a turma os resultados da avaliação que realizou.

Trabalho final

O trabalho final envolve a análise de requisitos, a proposta de um modelo conceptual, a implementação de parte de um protótipo e alguma avaliação de uma interface de utilizador, usando uma metodologia centrada no utilizador [19], que é apresentada na primeira aula prática

dedicada a este tema e que envolve a utilização de técnicas já estudadas nas aulas teóricas para a definição do perfil dos utilizadores alvo e análise contextual de tarefas [20].

Os alunos podem propor o tema do seu trabalho ou escolher um de dois enunciados apresentados pela autora, devendo, no primeiro caso, submeter uma página com a descrição dos objectivos e motivação da sua proposta. A maioria dos alunos acaba por escolher um dos enunciados propostos, mas todos os anos há grupos que apresentam a sua própria proposta, em geral relacionada com o projecto de licenciatura.

Neste último trabalho é fornecida apenas uma descrição muito geral da funcionalidade pretendida, por forma a que os alunos possam propor a funcionalidade específica com base no trabalho de levantamento de requisitos que venham a fazer.

Os alunos são livres de escolher a linguagem ou ferramenta que preferirem para implementar o protótipo, desde que façam uma demonstração na aula e forneçam o código e um executável juntamente com o relatório.

B. Outras actividades

No âmbito desta disciplina, os alunos têm também colaborado no trabalho de desenvolvimento ou de investigação de outras pessoas. Além de participarem como utilizadores em experiências controladas, tal como referido anteriormente, têm realizado avaliações heurísticas de *sites* e aplicações, bem como participado em testes de usabilidade, quer como observadores, quer como utilizadores. Exemplos destas colaborações são o teste de usabilidade e a experiência controlada descritos em [19] e [20]. A autora considera que estas colaborações têm sido muito positivas, já que permitem aos alunos, quer serem expostos à utilização dos métodos de avaliação que já conhecem em circunstâncias reais, quer melhor compreender a importância de saber lidar correctamente com as pessoas que colaboram neste tipo de avaliação.

C. Escrita dos relatórios

A escrita de um relatório é, para a maioria dos alunos, uma tarefa difícil. Sendo assim, explica-se com algum detalhe como o devem fazer, numa aula prática antes da data de entrega do primeiro relatório. Para apoiar os alunos nesta tarefa, é-lhes fornecido um documento onde se explica de forma simples quais as partes que devem constituir um relatório, sobre o que deve versar cada uma delas, bem como algumas directivas quanto ao estilo e formato e quanto à utilização de referências. Os alunos são ainda aconselhados a consultar uma compilação de artigos sobre como escrever e falar sobre tecnologia, publicada pelo IEEE [21].

Antes de entregarem os relatórios, os alunos devem dá-los a ler a outro grupo. Depois de entregues, os relatórios correspondentes ao trabalho sobre a avaliação heurística (o primeiro trabalho que é sujeito a avaliação) são lidos e comentados pela autora, sendo discutidos com cada grupo no decorrer das aulas seguintes. Este procedimento tem sido adoptado nos últimos anos e a autora tem notado uma melhoria significativa na qualidade dos relatórios.

VI. CONCLUSÕES

Neste artigo apresentaram-se os temas abordados nas aulas práticas e os trabalhos práticos realizados no âmbito da disciplina de Interfaces Humano-Computador, oferecida como opção, desde 1993/94, à Licenciatura em Engenharia Electrónica e de Telecomunicações (LEET) (e durante alguns anos ao Mestrado em Engenharia Electrónica e de Telecomunicações).

Como a Interacção Humano-Computador e a tecnologia avançam muito rapidamente, torna-se necessário um esforço contínuo para manter um conjunto de trabalhos práticos adequados.

Nos últimos anos, os alunos têm realizado dois tipos de trabalhos práticos; primeiro de avaliação de interfaces de utilizador e depois de projecto. Enquanto os primeiros trabalhos têm correspondido à avaliação de aplicações para plataformas *desktop*, *sites*, dispositivos *hand-held*, ou outros dispositivos interactivos (em geral de electrónica de consumo), os segundos têm incidido no projecto de interfaces de utilizador para aplicações Web ou *desktop*. No entanto, a autora considera actualmente a possibilidade de passar a incluir o projecto de interfaces de utilizador para PDA no trabalho de projecto, uma vez que a utilização deste tipo de plataformas se está a tornar cada vez mais importante, sendo a sua utilização certamente muito motivante para os alunos.

Nos primeiros anos em que a disciplina funcionou, realizaram-se, com resultados positivos, trabalhos em grupos de seis a oito alunos coordenados por um aluno de Mestrado; contudo esta abordagem, embora interessante, exige um controlo muito próximo do trabalho de cada elemento.

Finalmente, a autora acredita que a existência desta disciplina como opção na LEET é uma mais valia para os alunos, pois pode vir a ser muito útil aos que na sua futura actividade profissional venham a estar envolvidos no desenvolvimento não só de *software* interactivo, mas também de qualquer produto interactivo a desenvolver no âmbito da indústria automóvel, da electrónica de consumo ou das telecomunicações, entre outras.

REFERÊNCIAS

[1] Norman, D., *The Invisible Computer*, MIT Press, 1999

[2] ACM-SIGCHI, "Curricula for Human-Computer Interaction", *ACM - SIGCHI*, 1996 http://turing.acm.org/sigs/sigchi/cdg/cdg2.html (visitado em Outubro de 2004)

[3] Sommerville, I., *Software Engineering*, 6th ed., Addison Wesley, 2001

[4] Hu, S., "A Wholesome ECE Education", *IEEE Trans. on Education*, Vol. 46, N. 4, November, 2003, pp.444-451

[5] Evans, D., S. Goodnick, R. Rodel, "ECE Curriculum in 2013 and Beyond: Vision for a Metropolitan Public Reasearch University",

IEEE Transactions on Education, Vol. 46, N. 4, November, 2003, pp. 420-428

[6] McGettrick, A., M. Theys, D. Soldan, P. Srimam, "Computing Engineering Curriculum in the New Millenium", *IEEE Transactions on Education*, Vol. 46, N. 4, November, 2003, pp.456-462

[7] *ACM/IEEE* Joint Task Force, "Computing Curricula 2001-Computer Science", *ACM Journal of Educational Resources in Computing*, Vol. 1, N.3

[8] ACM/IEEE Joint Task Force, *Computing Curricula - Computing Engineering*, http://www.eng.auburn.edu/ece/CCEC (visitado em Outubro de 2004)

[9] Sousa Santos, B.,"Disciplinas da Área de Interacção Humano-Computador no Departamento de Electrónica e Telecomunicações da Universidade de Aveiro: Programa e principal bibliografia", *Actas do 2º Workshop Computação Gráfica e Multimédia na Educação CGME'03*, Outubro de 2003, Porto, pp.43-50

[10] Sousa Santos, B., "Disciplina de Interfaces Humano-Computador: Relatório sobre o programa, conteúdo e métodos de ensino teórico e prático", Universidade de Aveiro, 2004

[11] Dix, A., J. Finlay, G. Abowd, B. Russell. *Human Computer Interaction*, 2nd. ed., Prentice Hall, 1998

[12] Mayhew, D., *Principles and Guidelines in Software User Interface Design*, Prentice Hall, 1992

[13] Mayhew, D., *The Usability Engineering Lifecycle – A Practitioners Handbook for User Interface Design*, Prentice Hall, 1999

[14] Ferreira, C., B. Sousa Santos, M. E. Captivo, J. Climaco, C. C. Silva, "Multi-objective Location of Unwelcome or Central Facilities Involving Environmental Aspects - A prototype of a Decision Support System", *JORBEL, Belgian Journal of Operations Research, Statistics and Computer Science*, Vol.36, Nº 1-2, December 1996, pp.159-172

[15] Strong, G., New Directions in Human Computer Interaction Education, Research and Practice, 1994, http://www.sei.cmu.edu/community/hci/directions/ (visitado em Outubro de 2004)

[16] Nielsen, J., *Usability Engineering*, Academic Press, 1993

[17] Nielsen, J. http://www.useit.com/papers/ (visitado em Outubro de 2004)

[18] ANSI-354, *Common Industry Format Usability Test Reporting*, http://zing.ncsl.nist.gov/cifter/TheCD/cif/ readme.html (visitado em Outubro de 2004)

[19] Sousa Santos, B., F. Zamfir, C. Ferreira, Ó. Mealha, J. Nunes, "Visual Application for the Analysis of Web-Based Information Systems Usage: A Preliminary Usability Evaluation", *Proceedings of IEEE Conference on Information Visualization IV'04*, London, July 2004, pp. 812-818

[20] Silva, S., B. Sousa Santos, J. Madeira, C. Ferreira, "Comparing Three Methods for Simplifying Mesh Models of the Lungs: An Observer Test to Assess Perceived Quality", *Proceedings of SPIE Medical Imaging 2005*, San Diego, February 2005

[21] Beer, D., (ed.), *Writing and Speaking in the Technology Profession: A Practical Guide*, IEEE Press, 1992

Sistema de Informação Processual para a Provedoria de Justiça

Marco Fernandes, Miguel Alho, Pedro Almeida,
Joaquim Arnaldo Martins, Joaquim Sousa Pinto, Hélder Zagalo

Resumo **– Este artigo apresenta um Sistema de Informação Processual (SIP), desenvolvido pela Universidade de Aveiro para a Provedoria de Justiça. O sistema desenvolvido permite a pesquisa, recolha, representação e anotação de processos previamente digitalizados. No artigo, é descrita a estratégia adoptada no armazenamento e manipulação dos documentos digitalizados, na pesquisa de informação proveniente de múltiplos repositórios e no desenvolvimento de um sistema de anotações para documentos web.**

Abstract **–This article presents the project Sistema de Informação Processual, developed in the University of Aveiro for the Provedoria de Justiça. The system allows searching, retrieving, presenting and annotating the digital processes. In the paper, we describe the strategy adopted in the storage and manipulation of the digital documents, the search of information in multiple repositories and the development of a parallel system for the annotation of web documents.**

I. INTRODUÇÃO

Na sociedade de informação em que vivemos actualmente, as bibliotecas digitais assumem um papel de crescente relevo. As suas aplicações abrangem vários conteúdos, desde o papel (digitalização de documentos por uma organização/instituição), vídeo, fotografia, etc.

Com a sua implementação, pretende-se facilitar o processo de obtenção de informação por parte dos interessados, sendo este realizado de uma forma rápida e simples, em vários pontos de acesso. Por outro lado, procura-se criar um conjunto de facilidades, nomeadamente a facilidade de gestão e de organização, que dificilmente se poderia obter com os repositórios físicos originais (estejam eles em papel, vídeo analógico, etc.).

Apesar de oferecer um conjunto muito mais vasto e flexível de funcionalidades, as bibliotecas digitais colocam também vários problemas para quem as desenvolve. A segurança e a privacidade de informação obrigam a preocupações renovadas. Com efeito, numa lógica de acesso em vários pontos de uma rede (intranet ou, especialmente, internet) é imperativo reduzir ao mínimo as vulnerabilidades de segurança dos sistemas.

A decisão quanto ao formato de armazenamento é também um problema de grande relevo. Esta escolha depende de vários parâmetros, como o espaço ocupado, a perda de qualidade, o custo associado, o tempo de vida previsto para o formato e as ferramentas disponíveis para a manipulação dos documentos.

Requisitos

A Provedoria de Justiça tem, nos seus arquivos, mais de três milhões de documentos relativos aos processos existentes. Como é de fácil compreensão, a tarefa de obter um desses documentos para consulta pode ser inglória. Além disso, a gestão deste gigantesco repositório é um trabalho complexo.

Assim, a Universidade de Aveiro tinha de desenvolver um sistema que permitisse consultar os processos digitalizados e a informação existente na base de dados. Para facilitar o processo de consulta, deveriam ser desenvolvidas diversas funcionalidades de pesquisa. Finalmente, para permitir uma documentação mais completa e uma pesquisa mais orientada, deveria ser desenvolvido um sistema de anotação dos processos.

A aplicação web a desenvolver seria destinada à rede intranet da Provedoria, logo o número máximo de utilizadores simultâneos a que esta teria de responder ficaria abaixo de uma centena.

Modelo de 3 Camadas

Muitas aplicações actuais baseiam-se numa tecnologia cliente/servidor de duas camadas. Neste modelo, os dados são armazenados num servidor centralizado, que responde aos pedidos dos clientes que lhe acedem. Embora a gestão da informação seja simplificada, este modelo é muito pouco flexível e de difícil alteração posterior.

O desenvolvimento de uma aplicação complexa, seja ela web ou desktop, requer uma organização flexível e eficiente dos vários componentes que a constituem. O modelo mais comum para atingir este objectivo é composto por três camadas (*3-tier model*) [1]: dados, lógica e apresentação. Com esta arquitectura, conseguem-se melhorias substanciais na escalabilidade, robustez e reutilização do sistema. Alguns exemplos de bibliotecas digitais que usam este modelo são a biblioteca Alexandria [2] e o DSpace [3].

O acesso aos dados por parte da camada de apresentação é absolutamente transparente. Além disso, qualquer alteração na política de armazenamento e de acesso aos dados pode ser feita apenas nos componentes que lhe dizem respeito, sem ser necessário reconstruir todos os outros.

Por outro lado, o âmbito da aplicação pode ser alargado, já que outra qualquer aplicação pode ser construída sobre as camadas lógica e de dados já existentes.

Finalmente, com o acesso controlado aos dados, todos os detalhes de armazenamento e acesso aos mesmos podem ser encapsulados, aumentando consideravelmente a segurança do sistema.

Sistemas de Anotação

Os sistemas de anotação têm aplicações muito vastas e podem ser de grande utilidade. A grande funcionalidade destes sistemas é permitir inserir informações sobre o documento que se visualiza sem alterar o mesmo. Este conceito não é novo e algumas das mais populares aplicações de manipulação de documentos têm já ferramentas para inserir e manipular anotações: aplicações da família Microsoft Office [4], Adobe Acrobat [5], etc.

Embora esta funcionalidade esteja já bem implementada em aplicações desktop, ferramentas equivalentes para anotar documentos web estão ainda pouco desenvolvidas. Alguns exemplos são o Annotation Engine [6] e o MemoBook Notes [7].

No caso particular da Provedoria de Justiça, é de grande utilidade um mecanismo que complemente a informação disponível na base de dados ou nos documentos digitalizados. Estas anotações desde que feitas por pessoal qualificado, como é o caso em questão, são muito importantes para ajudar na classificação e posterior procura dos documentos, de uma forma mais selectiva e eficaz. Além disso permitem ir classificando os documentos de uma forma incremental, à medida que vão utilizados ou reutilizados, pois é impensável ter classificados em tempo razoável a enorme quantidade de documentos existente.

Num cenário intranet, como é o da Provedoria, uma aplicação deste tipo deve armazenar as anotações num repositório centralizado. Assim, ficam automaticamente disponíveis para os restantes utilizadores (se for essa a intenção), independentemente do ponto de acesso à rede.

V. ARQUITECTURA

Na Figura 1 está representada a arquitectura adoptada para o SIP. Como se pode verificar, o sistema foi desenvolvido de forma modular, o que permite efectuar alterações apenas nos componentes necessários.

Para aceder aos repositórios de informação, foram desenvolvidas três bibliotecas: ImageLibrary, DataLibrary e CNote. Sobre essas bibliotecas, dois web services foram criados – SIPws (geral) e espiritUs_ws (para o subsistema de anotações) – para disponibilizar os métodos necessários.

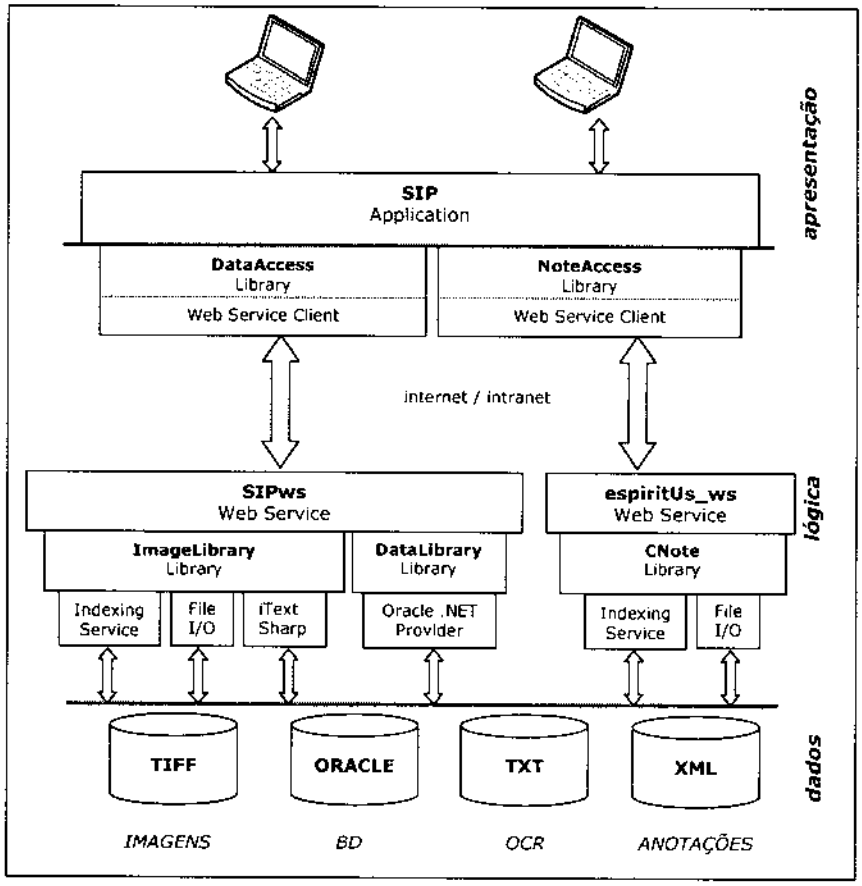


Figura 1 - Arquitectura geral do SIP

Finalmente, a aplicação web (SIP) acede de forma transparente aos dados utilizando os módulos DataAccess e NoteAccess, que por sua vez invocam os web services. Estes dois módulos permitem isolar a aplicação dos detalhes de acesso aos dados.

A. Camada de Dados

O sistema desenvolvido tem de manipular vários tipos de informação: os dados dos processos existentes na base de dados (Oracle) da Provedoria de Justiça, os documentos digitalizados desses mesmos processos, o texto extraído desses documentos por OCR (*Optical Character Recognition*) e as anotações efectuadas pelos utilizadores.

A.1 Repositórios de Informação

Na Provedoria de Justiça existia já uma base de dados com informação relativa aos processos existentes. Para implementar o Sistema de Informação Processual, foram criados três novos repositórios de informação: documentos digitalizados, documentos de texto extraído dos processos e documentos XML com as anotações aos processos.

A.2 Documentos Digitalizados

A escolha de um formato para armazenamento dos documentos digitalizados foi feita tendo em conta a conservação de uma alta qualidade de imagem sem exigir demasiada capacidade de armazenamento. Depois de uma análise dos vários formatos existentes, foi definido que os documentos seriam digitalizados para o formato TIFF (*Tag Image File Format*) [8], com compressão CCITT (*Comité Consultatif International Télèphonique et Télègraphique*) Grupo 4 e a uma resolução de 300 dpi. O formato é bastante eficaz na conservação da informação, permitindo quer a visualização quer a impressão de alta qualidade e ainda a possibilidade da transcodificação para outros formatos. Visto que os documentos são (tipicamente) a preto e branco, o tamanho final do ficheiro é bastante reduzido (cerca de 40KB) apesar das suas dimensões (2481x3512).

A distinção das páginas é feita através do nome do ficheiro que inclui a informação do número da página e também o número e ano do processo. Esta designação única em conjunto com o serviço de indexação permite que a estrutura dos directórios no repositório seja arbitrária, já que o serviço encarrega-se de determinar a sua localização, tornando o acesso aos documentos transparente. O serviço de indexação foi implementado utilizando o Indexing Service da Microsoft.

O Indexing Service (IS) [9] é um serviço do Windows 2000 e versões posteriores que extrai conteúdo de documentos e cria um catálogo para permitir uma pesquisa mais rápida e eficiente. Por defeito, o IS filtra documentos Office, HTML, mensagens MIME e ficheiros de texto e indexa informação específica (autor, conteúdo, etc.). Para todos os restantes ficheiros, apenas são filtradas propriedades genéricas (data, nome e localização do ficheiro, etc.)

A.3 Documentos OCR

O texto digitalizado dos documentos dos processos é armazenado em ficheiros de texto simples. O formato é universalmente aceite e armazenado no mais reduzido espaço possível. Assim, a sua transferência é bastante rápida. Tal como no caso dos documentos digitalizados (imagens), o nome do ficheiro serve para identificar a que página de determinado processo pertence o texto.

A.4 Anotações

Ao contrário do que acontece nos repositórios anteriores, não é utilizado apenas o nome do ficheiro para determinar a que processo (e, eventualmente, a que documento) pertence um conjunto de anotações. Como se explicará na secção B.3, um ficheiro de anotações fica associado ao URL (*Uniform Resource Locator*) do documento exibido no *browser* do utilizador. Concretamente, a estrutura de directórios e o nome dos ficheiros no repositório reflectem o endereço de um determinado documento web.

O armazenamento das anotações é feito no formato XML. O XML (Extensible Markup Language) [10] é um formato de texto simples e flexível derivado do SGML (Standard Generalized Markup Language) [11]. Este formato permite armazenar informação em blocos indentificados com marcadores. A sua utilização traz grandes vantagens, nomeadamente: flexibilidade estrutural, nomeação dos marcadores de forma personalizada e intuitiva e existência cada vez mais relevante de ferramentas de manipulação de XML.

Um documento XML de anotações tem um formato semelhante ao que se apresenta de seguida.

```
<?xml?>
<notes url="">
 <query txt="">
  <note id="" userid="" private="" date="">
  </note>
 </query>
</notes>
```

Esta estrutura de informação armazena o URL do documento consultado, a identificação do utilizador, a indicação se a anotação é privada ou pública, a data (e hora) em que foi realizada a notação e a anotação propriamente dita.

Como este componente de anotações ainda está em fase de desenvolvimento, optou-se por utilizar o XML como formato de armazenamento das anotações devido à sua flexibilidade. Se, numa fase posterior, se pretender extender a funcionalidade deste componente, nomeadamente ao permitir criar *threads* de anotações (encadeamento), facilmente se alterará a estrutura XML de

suporte ao armazenamento da informação, ao contrário do que sucederia com bases de dados relacionais.

B. Camada Lógica

A camada lógica é responsável por implementar políticas de acesso e manipulação dos dados. Nesta camada, implementou-se um conjunto de funcionalidades que visam aumentar a *performance* do sistema e a transformação de dados. Como forma de isolar a camada lógica da camada de apresentação, utilizaram-se Web Services.

B.1 Web Services

Os Web Services [12] são uma das principais ferramentas para implementar arquitecturas distribuídas em aplicações web. São unidades individuais de código, que permitem a partilha de dados entre aplicações em diferentes máquinas, mesmo em plataformas distintas. Além disso, o seu formato de transferência de informação é o XML, cuja utilização, como se viu na secção anterior, tem grandes vantagens.

B.2 Manipulação de Documentos Digitalizados

O formato TIFF não é suportado para representação de imagens nos *browsers* típicos. Assim, para permitir a apresentação dos processos via web, é necessário um processo intermédio.

A solução adoptada podia ter passado pela instalação de um *plug-in* (por exemplo, o Quicktime [13] da Apple) que permitisse a visualização de imagens do tipo TIFF. Embora esta solução funcione, exige que todos os utilizadores possuam o componente. Além disso, o tempo de carregamento do *plug-in* torna a experiência de navegação na aplicação pouco agradável.

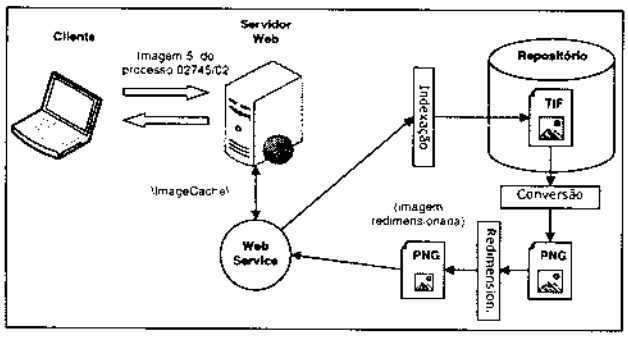


Figura 2 - Consulta de uma página de um processo

Assim, optou-se por efectuar, *on-demand* e em tempo real, a conversão dos documentos TIFF para o formato PNG (*Portable Network Graphics*) [14]. Este é um dos formatos suportados pelos *browsers* com uma das melhores relações qualidade/tamanho. Além da conversão, é feito ainda um redimensionamento do tamanho para 680x962. Desta forma, a informação transmitida entre a camada de dados e a de apresentação é mais reduzida. Por outro lado, um redimensionamento prévio da imagem para o tamanho em que vai ser exibido torna-a mais legível do que quando o mesmo é feito pelos browsers. O processo é demonstrado na Figura 2.

Como se pode observar na Figura 2, cada documento convertido fica armazenado numa cache de imagens de tamanho configurável. Desta forma, se o mesmo ou outro utilizador voltar a requisitar uma imagem já existente na cache, o processo de conversão e redimensionamento não é feito. Quando a cache fica preenchida, o sistema determina a imagem mais antiga e remove-a. Tendo em conta que, de acordo com os elementos da Provedoria, num determinado período temporal existe um padrão que indica a consulta sistemática dos mesmos processos, a utilização de uma cache (mesmo que de tamanho modesto) revela-se uma mais-valia para o sistema.

Quanto aos processos digitalizados, o sistema de informação processual apresenta ainda outra funcionalidade – a impressão completa ou parcial de um processo. Para atingir esse objectivo, foi criada uma biblioteca que reúne todos os documentos solicitados (TIFF) num único PDF (*Portable Document Format*) [15], que é posteriormente enviado para o cliente. Em cada página, para além da imagem digitalizada, pode ser ainda colocado um cabeçalho com o número da página e do processo. O processo está exemplificado na Figura 3.

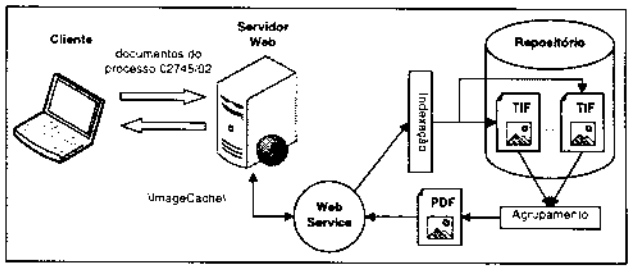


Figura 3 - Criação de PDF para impressão

Tal como no caso dos PNG, os documentos PDF são armazenados numa cache do sistema (no mesmo espaço físico que a de PNG). Contudo, esta cache terá, previsivelmente, uma utilização menos eficaz que a anterior, já que, para cada processo, cada utilizador pode gerar uma colecção diferente de documentos.

B.3 Gestão das Anotações

O sistema de anotação desenvolvido armazenada as anotações de acordo com o URL da página que está a ser visualizada. Por exemplo, se o sistema for utilizado para anotar a página http://www.ieeta.pt/A/B/pagina.htm, a anotação é armazenada na directoria [Repositorio]\A\B\ com o nome pagina.htm.xml.

No caso do sistema da Provedoria, para visualizar a ficha do processo 1234 de 2001, é utilizado um endereço semelhante a http://[*Servidor*]/processo.aspx?n=01234/01. Caso fosse este o endereço utilizador para realizar as anotações, estas seriam armazenadas no repositório de anotações no ficheiro processo.aspx.xml no directório

[Repositorio]\[Servidor]. A informação que aparece a seguir ao ponto de interrogação seria armazenada no elemento *query* do documento XML, conforme apresentado na secção A.4.

Esta forma de armazenar as anotações, embora genérica, tornaria o sistema composto por ficheiros em reduzido número mas de grande dimensão, já que todas as anotações a fichas de processos seriam armazenadas neste ficheiro. Posteriormente, utilizar-se-ia o elemento *query* para determinar quais as anotações de um determinado processo. Estes ficheiros, de grande dimensão, poderiam levantar problemas no processo de pesquisa e manipulação de anotações.

Para ultrapassar este problema, foi desenvolvido um filtro ISAPI (*Internet Information Application Protocol Interface*) [16], que se encarrega de transformar o URL de apresentação dos processos para o seguinte formato: http://[*Servidor*]/processos/2001/1234. Ao anotar esta página, é armazenado um ficheiro no directório [Repositorio]\[*Servidor*]\documentos\2001\1234.xml no servidor de anotações. Deste modo, as anotações relativas a cada processo ficam armazenadas em ficheiros separados.

B.4 Funcionalidades de Pesquisa

A pesquisa incide, essencialmente, na base de dados, no texto OCR e no texto anotado. Quanto à base de dados, foram escolhidos alguns campos da mesma, considerados relevantes pelos elementos da Provedoria, para efectuar pesquisas. Quanto à pesquisa no texto OCR e nas anotações foi utilizado o IS, embora de formas distintas. No caso dos processos digitalizados em TIFF, é utilizado o serviço de indexação para determinar quais os ficheiros (através do seu nome) associados a um determinado processo.

No caso do texto associado a esses documentos (OCR), a metodologia seguida é um pouco diferente. Com efeito, estes documentos são essencialmente utilizados para efectuar pesquisas por texto livre no seu conteúdo, sendo o número do processo determinado posteriormente a partir do nome do ficheiro.

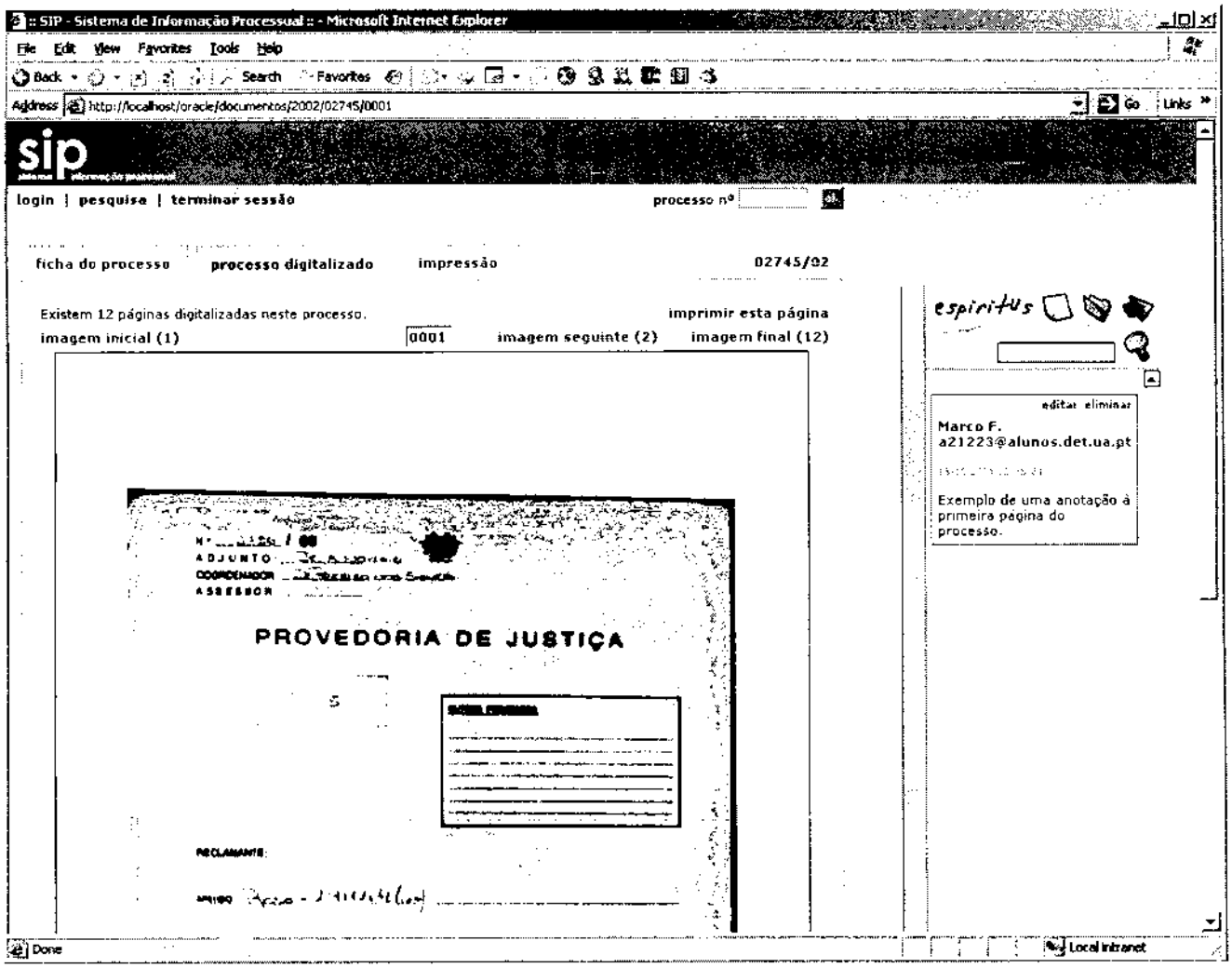


Figura 4 - Interface do SIP

Como o IS não tem, por defeito, um filtro para documentos XML, foi necessário instalar um filtro XML (IFilter , da Quilogic [17]) para indexar as anotações.

C. Camada de Apresentação

Como se pode observar na Figura 4, a informação é disponibilizada aos utilizadores sob a forma de páginas web. Existe uma página de pesquisa onde se podem procurar processos pelos critérios descritos acima. Para cada processo, é disponibilizada a ficha do mesmo (proveniente da base de dados) e é permitido consultar todas as páginas digitalizadas (no formato PNG). Os utilizadores podem igualmente gerar PDFs de todo o processo ou de apenas de algumas páginas para salvaguardar localmente ou imprimir.

Quanto às anotações, foi desenvolvido um componente que se pode colocar em qualquer página onde se pretenda que existam anotações (no caso da Provedoria, foi colocado na página dos processos). Cada utilizador, depois de autenticar-se, pode fazer anotações a um processo no seu todo ou a uma página em concreto. Esta anotação pode ser pública (livre para consulta) ou privada (visível apenas para o autor). Pode ainda obter a listagem de todas as suas anotações e efectuar uma pesquisa nas anotações existentes (suas e de outros, quando públicas).

VI. CONCLUSÕES

O sistema desenvolvido cumpre as funcionalidades essenciais exigidas, integrando na página web informação de fontes heterogéneas.

Encontrou-se uma solução que permite conservar os documentos digitalizados num formato com alta resolução – o TIFF. Tanto a conversão para PNG como o processo de junção de várias imagens num PDF são feitas de forma relativamente célere. Além disso, a existência de uma cache de documentos torna o sistema notoriamente mais eficiente.

Por outro lado, graças à utilização do sistema de indexação, o acesso aos repositórios é feito de forma completamente transparente para a aplicação, já que esta não necessita de saber a localização de cada documento.

Finalmente, a arquitectura do sistema desenvolvido é completamente modular, pelo que alterações posteriores podem ser executadas de forma eficaz.

REFERÊNCIAS

[1] Jupitermedia Corporation, *three-tier – Webopedia.com*, 2004, http://winplanet.webopedia.com/TERM/T/three_tier.html

[2] University of California, *Alexandria Digital Library Project*, 2004, www.alexandria.ucsb.edu

[3] MIT Libraries & Hewlett-Packard, *DSpace Federation*, 2003, http://dspace.org

[4] Microsoft Corporation, *Microsoft Office System Informação de Produto*, 2004, http://www.microsoft.com/portugal/office

[5] Adobe Systems, *Adobe Acrobat 6.0 Professional*, 2004, http://www.adobe.com/products/acrobatpro/overview.html

[6] Berkman Center for Internet & Society – Harvard, *Annotation Engine*,2004, http://cyber.law.harvard.edu/projects/annotate.html

[7] Obout Inc., *MemoBook Notes*, 2003, http://www.memobook.com

[8] O'Reilly & Associates, Inc, *GFF Format Summary: TIFF*, 1996, http://netghost.narod.ru/gff/graphics/summary/tiff.htm

[9] Microsoft Corporation, *What is Indexing Service*, 2003, http://msdn.microsoft.com/library/en-us/indexsrv/html/ixintro_0311.asp

[10] World Wide Web Consortium, *Extensible Markup Language (XML) 1.0 (Third Edition)*, 2004, http://www.w3.org/TR/REC-xml

[11] ISO 8879:1986, *Standard Generalized Markup Language (SGML)*, 2001, http://www.iso.org

[12] World Wide Web Consortium, *Web Services*, 2004, http://www.w3.org/TR/ws-arch

[13] Apple Computer, *Apple – Quicktime*, 2004, http://www.apple.com/quicktime

[14] Greg Roelofs, PNG (Portable Network Graphics) Home Site, 2004, http://www.libpng.org/pub/png

[15] Adobe Systems, *What is Adobe PDF?*, 2004, http://www.adobe.com/products/acrobat/adobepdf.html

[16] Microsoft Corporation, *ISAPI Filter Overview*, 2004, http://msdn.microsoft.com/library/en-us/iissdk/iis/isapi_filter_overview.asp

[17] Quilogic, *XML IFilter for indexing XML files*, 2003, http://www.quilogic.cc/ifilter.htm

Integração de Informação na equipa de Futebol Robótico CAMBADA

Paulo Bartolomeu, Luis Seabra Lopes, Nuno Lau, Armando Pinho, Luis Almeida

Abstract – **This article describes the software architecture of a *CAMBADA* autonomous agent, focusing on the information integration mechanisms. The system requirements and the developed solutions for data structures and information fusion are also discussed. Additionally, some error detection and correction mechanisms used in the *CAMBADA* robots are presented.**

Keywords – **Mobile Robots, Multi-Agent Systems, Sensor Fusion, Cooperative Sensing, Robotic Soccer**

Resumo – **Este artigo descreve a arquitectura de software de um agente autónomo *CAMBADA* focando os mecanismos de integração de informação. Discutem-se em particular os requisitos do sistema e as soluções encontradas ao nível das estruturas de dados e da fusão de informação. Adicionalmente apresentam-se alguns dos mecanismos de detecção e correcção de erros usados nos robôs *CAMBADA*.**

Palavras chave – **Robótica Móvel, Sistemas Multi-agente, Fusão Sensorial, Percepção em Equipa, Futebol Robótico**

I. INTRODUÇÃO

O projecto CAMBADA (*Cooperative Autonomous Mobile roBots with Advanced Distributed Architecture*) [1] [1] tem como objectivo a construção de uma equipa de futebol robótico F-2000 (*Middle-Size League*) para participação no *RoboCup* [2]. Este projecto é actualmente o mais importante da Actividade Transversal em Robótica Inteligente (ATRI) [3] do IEETA [4] envolvendo a quase totalidade dos seus elementos. De entre as inúmeras áreas científicas que contribuíram para a sua realização, a Inteligência Artificial assume neste artigo especial relevo.

O *RoboCup* é uma iniciativa internacional que tem por objectivo promover o desenvolvimento de uma equipa de futebol robótico que em 2050 seja capaz de vencer um jogo à equipa humana campeã do mundo sob as regras da FIFA [5]. Esta iniciativa possui várias classes de competição, entre elas a *Middle-Size League* (MSL). Esta classe é caracterizada por robôs de tamanho médio (30cm x 30cm $\leq$ área ocupada $\leq$ 50cm x 50cm, 40cm $\leq$ altura $\leq$ 80cm e peso $\leq$ 40kg) com sensores incorporados que jogam futebol num campo de dimensões entre 8m x 6m e 16m x 12m. Algumas convenções possibilitam a identificação de objectos pelo padrão de cor. O terreno de jogo é verde, as linhas marcadas a branco, e as balizas possuem cores distintas, uma amarela e outra azul. Os postes possuem o padrão de cor correspondente ao lado do campo em que se encontram (por exemplo, do lado da baliza amarela os postes são 2/3 amarelos e 1/3 azuis, sendo que este 1/3 azul deve corresponder à faixa central do poste). A bola é cor de laranja.

Cada jogo é dividido em dois períodos de 10 minutos com um intervalo de também 10 minutos. Cada equipa pode jogar com um máximo de 6 robôs (guarda-redes incluído) sendo estes genericamente de cor preta exceptuado a faixa colorida identificativa da equipa (magenta ou azul claro). Os robôs devem jogar em equipa podendo utilizar a tecnologias sem fios, nomeadamente Wi-Fi [7], para comunicarem entre si.

Através de uma estação de controlo electrónica (*referee box*), a equipa de arbitragem comunica com as estações base de cada uma das equipas para dar instruções acerca do estado do jogo.

A estação base é um agente de software da equipa mas que não está instalado em nenhum robô. Este agente pode comunicar com os restantes informado-os das alterações do estado do jogo recebidas da *referee box* e adicionalmente pode desempenhar o papel de "treinador" dando instruções ao nível da estratégia de jogo da sua equipa. No decorrer de uma partida existem diversos cenários em que a estação base pode intervir em resposta a um comando da estação de controlo. Os cenários mais comuns são: inicio ou fim de jogo, retirada ou re-entrada de um jogador, marcação de uma grande penalidade ou canto, etc.

As características da liga MSL relativamente às condições do jogo, do campo e dos jogadores condicionaram a arquitectura de informação dos agentes da equipa CAMBADA.

No resto do artigo, começa-se por apresentar sucintamente a equipa CAMBADA, quer a nível de hardware, quer a nível da arquitectura de software. Em seguida, na secção III, apresenta-se a estratégia de partilha de informação entre agentes e entre processos no mesmo agente, a qual se baseia numa Base de Dados de Tempo-Real. A secção IV apresenta e detalha as estruturas de dados utilizadas. A secção V constitui o tema central deste artigo. Nesta secção discutem-se os problemas identificados ao nível da integração de informação de um agente e apresentam-se as soluções desenvolvidas para os ultrapassar. Finalmente, a última secção conclui este artigo apresentando algumas linhas de trabalho futuro.

II. A EQUIPA CAMBADA

A. Estrutura Mecânica e Sensorial dos Robôs

A estrutura mecânica de um robô da equipa CAMBADA é modular, organizando-se em três camadas. A camada inferior (Figura 1) inclui as rodas, o *kicker*, os motores e as baterias. Tudo isto está montado numa placa de alumínio de formato circular, com $440mm$ de diâmetro, e com cortes que permitem o direccionamento da bola (corte semi-circular

[1] Projecto POSI/ROBO/43908/2002 financiado pela FCT, Fundação para a Ciência e a Tecnologia, e pelo FEDER.

no eixo Y) e a fixação das rodas.

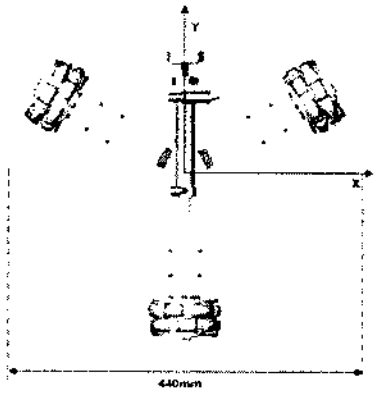


Figure 1 - Base do robô CAMBADA

Os robôs podem-se mover em todas as direcções, ou seja, possuem movimento holonómico. Este movimento é suportado pela utilização de rodas especiais (*OmniWheels*) que permitem movimentos segundo dois eixos em simultâneo. As rodas são dispostas de forma a fazerem um ângulo de 120 graus entre si. Os motores possuem *encoders* permitindo obter o número de rotações que cada roda realiza. A odometria é realizada no baixo-nível (placas de microcontrolador) recorrendo a leituras dos *encoders* e posterior processamento.

A segunda camada agrega toda a electrónica de controlo motor, a qual consiste num conjunto de microcontroladores interligados por uma rede CAN. Dado o desenho modular dos robôs, esta camada pode facilmente ser substituída por outra igual.

A terceira camada inclui um computador pessoal, onde é executado todo o software de controlo de alto nível e integração de informação, bem como as câmaras (*webcam*) que compõem o sistema de visão.

O sistema de visão artificial é composto por duas câmaras. A câmara omnidireccional, direccionada para baixo, é usada para possibilitar a visão de objectos em qualquer posição próxima do robô. A câmara frontal previligia a visão numa determinada direcção, permitindo ver mais longe nessa direcção. Esta câmara está orientada segundo o eixo Y do robô, ficando esta a ser a frente do robô. Com o mesmo alinhamento foi colocado o *kicker*.

A Figura 2 mostra um robô da equipa CAMBADA inteiramente funcional.

Observe-se ainda na Figura 2 o *laptop* que incorpora o robô. Este dispositivo computacional é responsável pelo processamento da informação de alto-nível e pelo comportamento global do robô.

B. Fluxo de Informação Global

A Figura 3 apresenta um diagrama ilustrativo do fluxo de informação de alto nível num jogador CAMBADA. As setas a cheio indicam fluxo de controlo e as setas a tracejado indicam fluxo de informação.

A informação produzida pelos processos das câmaras (*Visão Omnidireccional* e *Visão Frontal*) inicia o ciclo de processamento. Esta informação descreve as posições relativas (ao próprio robô) dos objectos identificáveis pelas

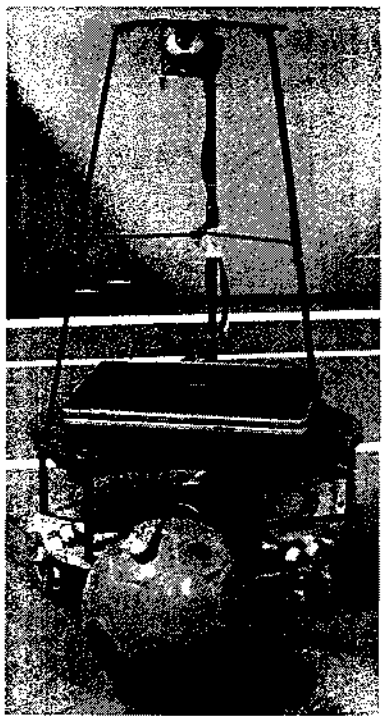


Figure 2 - Robô CAMBADA

câmaras. Os processos da visão disponibilizam esta informação para o processo principal escrevendo-a numa área de memória designada de *Área Local*. Esta área de memória é usada como mecanismo de comunicação entre os processos residentes no robô (processos da visão e processo principal).

O processo principal lê informação da área de memória local e usa-a para calcular, entre outros parâmetros, a sua posição absoluta no campo, a posição da bola, etc. Este procedimento designa-se por *Integração de Informação* e é caracterizado pela fusão de dados sensoriais que resulta em informação de natureza posicional.

A informação obtida da integração é armazenada (*Actualização do Estado do Mundo*) na área de memória partilhada reservada para o próprio agente. Esta área é ilustrada na Figura 3 por duas zonas de memória designadas de *Estado do Mundo*. O *Estado do Mundo* representado a cheio refere-se ao próprio agente enquanto que o *Estado do Mundo* representado a tracejado representa os estados do mundo dos outros agentes. A existência da área de memória partilhada visa possibilitar a comunicação entre agentes. Como resultado, esta área é acessível a todos os agentes para leitura e apenas ao próprio para escrita no *Estado do Mundo* correspondente.

Em seguida, o processo principal realiza a *Integração de Informação da Equipa*. Este procedimento é responsável pela fusão das diferentes visões de jogo dos robôs. O processo de fusão pretende produzir informação (localização dos robôs, da bola e das referências do campo) consistente com as diversas visões de jogo.

Finalmente e com toda a informação disponível, o processo principal decide qual o comportamento a realizar (*Decisão*), executando-o (*Execução de Comportamentos*). O algoritmo de decisão, que tem por base uma máquina de

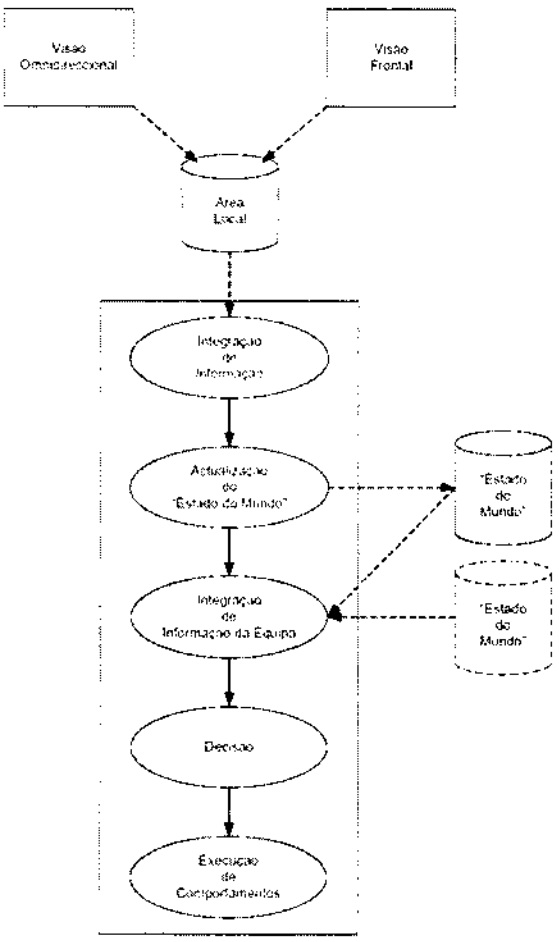


Figure 3 - Fluxo de informação Global

estados, toma em consideração não só informação proveniente das integrações prévias como também considera outros factores importantes, nomeadamente a função do robô na equipa e o estado do jogo.

Os procedimentos apresentados são executados ciclicamente e de acordo com o ritmo a que as câmaras produzem informação.

III. PARTILHA DE INFORMAÇÃO ENTRE PROCESSOS E AGENTES

Do ponto de vista da arquitectura de software, a equipa *CAMBADA* é um conjunto de agentes (os jogadores e a estação base) e o comportamento de cada agente resulta da execução de um conjunto de processos de software, nomeadamente processos percepcionais, decisórios e de controlo. Assim, é importante ter formas de partilhar informação entre processos, no mesmo agente, bem como entre os vários agentes.

O mecanismo de partilha de informação desenvolvido baseia-se numa base de dados de tempo-real distribuída (RTDB) [8], [9]. Esta base de dados reside em todos os robôs e a informação presente em cada uma delas é sincronizada com as restantes por intermédio de um algoritmo próprio.

A organização global das estruturas de dados e a sua organização individual numa base de dados constitui um factor de qualidade que tem reflexos directos na eficiência do armazenamento e na manipulação da informação.

A organização global da base de dados tempo-real desenvolvida é apresentada na Figura 4.

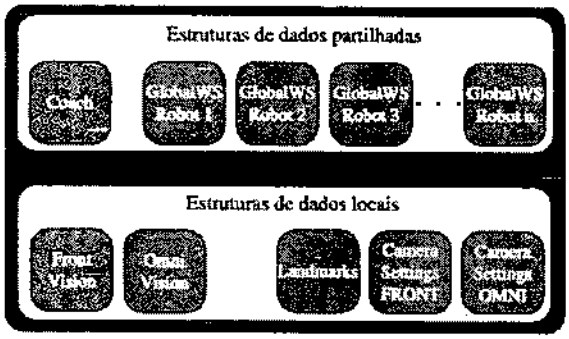


Figure 4 - Organização global das estruturas de dados

Como se pode observar existe uma separação clara entre a informação que deverá ser partilhada com os restantes agentes e a informação local do agente robótico.

A. *Área Local*

A área local da RTDB destina-se apenas a blocos de dados que não sejam passíveis de serem partilhados com os restantes robôs. Neste contexto, encontram-se os parâmetros das câmaras (*Camera Settings FRONT* e *Camera Settings OMNI*), as posições de marcas fixas no campo (*Landmarks*) e finalmente os dados obtidos a partir das câmaras (*Front Vision* e *Omni Vision*).

Os parâmetros das câmaras e a posição de marcas fixas no campo residem na área Local da RTDB para facilitar a sua manipulação. Uma vez que estes parâmetros e estas posições são inicializadas no primeiro ciclo de funcionamento de cada robô permanecendo posteriormente inalterados, não foi considerada relevante a sua difusão pelos restantes robôs. Adicionalmente, os parâmetros das câmaras são locais por definição uma vez que modelam características físicas das câmaras montadas no robô.

A informação produzida pelas câmaras é local uma vez que apenas o processo principal possui informação adicional que possibilita a sua integração. Os processos da visão comunicam com o processo principal através da área local da RTDB. A utilização da RTDB neste contexto permite uniformizar a transferência de informação entre processos locais.

B. *Área Partilhada*

A área partilhada da RTDB destina-se a informação não só relevante para o próprio robô mas também para a sua equipa. A informação constante desta área é difundida periodicamente por todos os elementos da equipa estabelecendo uma plataforma de conhecimento comum do seu "mundo".

Quando começa, ou no decorrer de um encontro de futebol robótico, a estação base instrui os elementos da equipa para assumirem um determinado comportamento. A comunicação das instruções é realizada ao nível da RTDB recorrendo à estrutura *Coach* aí residente.

Esta estrutura apenas pode ser escrita pela estação base mas pode ser lida por todos os elementos da equipa.

A informação constante da classe *GlobalWS* caracteriza a localização e o estado de um robô. A partilha da informação potencia a tomada de decisão de forma mais eficaz por qualquer elemento da equipa. Desta forma, os robôs dispõem também das visões individuais de cada um dos robôs em jogo podendo "decidir" de forma mais "esclarecida".

IV. ESTRUTURAS DE DADOS

Esta secção apresenta as estruturas desenvolvidas, referindo as motivações e os requisitos de cada uma. Para maior clareza, as classes desenvolvidas são apresentadas em notação UML (*Unified Modeling Language*) [10]-[12].

A. O "Estado do Mundo"

O "Estado do Mundo" é uma estrutura de dados que modela a organização da informação essencial a um agente robótico. A organização desta informação foi concebida tomando como objectivos a separação semântica dos diversos elementos que constituem o "Estado do Mundo", a reutilização de estruturas de dados genéricas (Vectores, etc.) e a adequação do modelo ao problema.

Após um estudo preliminar identificaram-se os seguintes elementos como essenciais ao "Estado do Mundo":

- localização, orientação e velocidade do robô e dos colegas de equipa no campo;
- localização, orientação e velocidade dos adversários no campo;
- localização e velocidade da bola no campo;
- parâmetros específicos do robô (por exemplo a sua função na equipa);
- e finalmente, parâmetros específicos da equipa (por exemplo a cor da equipa).

A localização das balizas, dos postes ou de quaisquer outros elementos estáticos no campo foi considerada irrelevante para o "Estado do Mundo" visto que os mesmos são conhecidos *a priori* e não se alteram no decorrer de uma partida.

Os elementos identificados resultaram em classes que integram a classe *GlobalWS* representada na Figura 5. O "Estado do Mundo" de um robô é assim constituído por um conjunto de objectos (Robôs, Bola e parâmetros adicionais do robô e da equipa) que modelam separadamente objectos físicos e parâmetros funcionais.

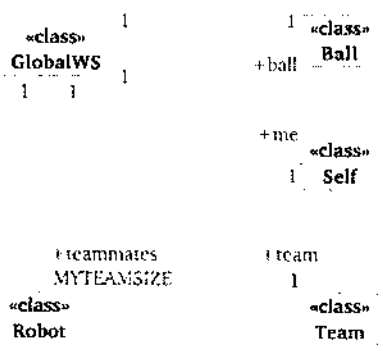


Figure 5 - "Estado do Mundo"

O número de atributos da classe *Robot* que uma dada classe *GlobalWS* possui é dado pelo somatório dos elementos de cada equipa[2]. Cada "Estado do Mundo" possui ainda informação relativa aos parâmetros adicionais para o próprio robô (*Self*), aos parâmetros da equipa (*Team*) e informação relativa à bola (*Ball*).

As subsecções seguintes descrevem em detalhe as classes que compõem a classe *GlobalWS*.

A.1 A classe Object e suas derivadas

A classe *Object* foi desenvolvida com o objectivo de modelar as características de um objecto real, para que outros objectos reais como robôs e bola pudessem possuir estruturas de base semelhantes diferindo apenas na sua especificidade.

A classe *Object* é constituída por diversos elementos de informação que permitem caracterizar parcialmente um objecto genérico, nomeadamente:

- posição absoluta do objecto;
- posição relativa do objecto (ao próprio robô);
- o objecto encontra-se visível?
- o objecto encontra-se visível com índice de confiança elevado?
- tempo de vida da informação.

Conforme ilustrado na Figura 6, as classes *Ball* e *Robot* derivam da classe *MobileObject*, que por sua vez deriva da classe *Object*. Ou seja, um objecto móvel (*MobileObject*) é um objecto que pode ter uma determinada velocidade. A bola é um objecto móvel. Um robô é um objecto móvel com uma orientação.

Observam-se ainda na mesma figura duas variáveis booleanas *see* e *seeXY*. Estas variáveis permitem determinar se o objecto se encontra visível com confiança elevada, com confiança moderada ou não se encontra visível de todo.

A existência de duas variáveis booleanas *see* e *seeXY* deve-se a que a visão dos robôs não apresenta os mesmos índices de confiança em todas as direcções. As lentes usadas no sistema de visão possuem um elevado nível de distorção que é parcialmente suprimido usando correcção por *software*. Esta correcção conduz a uma compressão da imagem que resulta em duas áreas distintas: área corrigida e área não-corrigida. Os objectos visíveis nesta última apresentam um grau de confiança diminuto por oposição aos objectos visíveis na área corrigida. Assim, a variável booleana *seeXY* assinala a visibilidade do objecto na área corrigida e a variável booleana *see* assinala a visibilidade do objecto na imagem (independentemente da área onde este se encontra).

A Figura 6 mostra adicionalmente os atributos que compõem as classes *Rotation* e *Vector*.

[2]A equipa do robô possui *MYTEAMSIZE* elementos (incluíndo ele próprio) e a equipa adversária *THEIRTEAMSIZE* elementos.

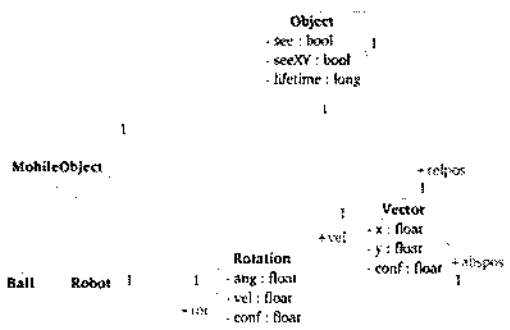


Figure 6 - Classe *Object* e suas derivadas

A.2 A classe Self

A classe *Self* (Figura 7) permite armazenar informação adicional do robô que não seja adequada para constar da classe *Robot*. Um robô possui um número que o distingue dos restantes e um "papel" na equipa que tipicamente está associado ao seu comportamento em campo. Esta classe integra dois atributos das classes *See* e *Distance* e uma variável para contabilizar a "idade" da informação. A classe *See* é apenas um conjunto de variáveis booleanas indicando a visibilidade das balizas e a classe *Distance* armazena a distância total que o robô percorreu desde que foi ligado.

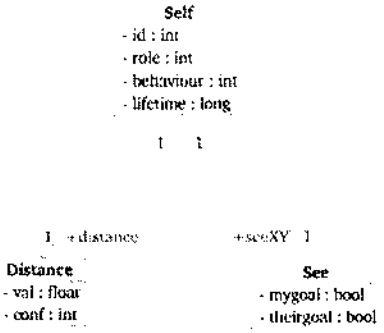


Figure 7 - Classe *Self*

A.3 A classe Team

A classe *Team* (Figura 8) armazena apenas informação relativa à equipa. Nestas circunstâncias identificou-se apenas a cor da equipa como atributo. Em fases de desenvolvimento posteriores outros atributos poderão surgir.

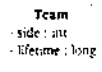


Figure 8 - Classe *Team*

A.4 A classe GlobalWS

A classe *GlobalWS* caracteriza o estado do mundo e agrega as classes anteriormente apresentadas. Esta classe possui adicionalmente informação sobre o número de ciclos de controlo executados, número de inconsistências ocorridas[3], estado do jogo no ciclo de controlo actual, estado do jogo no ciclo de controlo anterior e várias variáveis booleanas que são descritas em seguida:

- *base_station* - Indica se este agente é a estação base (verdadeiro/falso);
- *station_active* - Indica o estado da estação base (verdadeiro/falso);
- *ball_seen* - Indica se a bola foi vista no actual ciclo de controlo (verdadeiro/falso);
- *running* - Indica se "este" robô está em operação (verdadeiro/falso);
- *behaviour_changed* - Indica se o comportamento "deste" robô foi alterado no ciclo de controlo actual (verdadeiro/falso);
- *teammates_active[MYTEAMSIZE]* - *Array* de variáveis booleanas que assinala os agentes que estão activos. Entende-se por agentes activos aqueles cuja variável correspondente do *array* seja verdadeira.

A Figura 9 ilustra a classe *GlobalWS*.

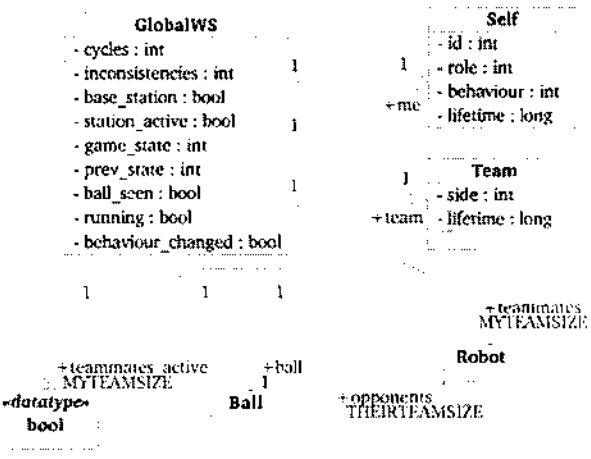


Figure 9 - Classe *GlobalWS* em detalhe

A informação que a classe *GlobalWS* agrega permite caracterizar a visão do mundo que um dado agente robótico possui. Esta informação é posteriormente usada no processo de decisão comportamental do robô.

B. O Treinador - Coach

A classe *Coach* é a estrutura de dados que modela a organização da informação correspondente ao treinador. Esta classe é usada para "transportar" informação de estado de jogo e parâmetros de início (ou re-início) da partida.

Como se pode observar pela Figura 10 a classe *Coach* possui adicionalmente informação sobre a cor da equipa, estado da estação base (activa/inactiva), estado (activo/inactivo) e "papel" de cada um dos robôs que participam na partida.

[3]Consideram-se como inconsistências situações anómalas, i.e., situações originadas por medidas sensoriais inconsistentes. Um exemplo de uma situação destas é o robô determinar que a bola se encontra fora do terreno de jogo a uma distância de 100m deste.

Figure 10 - Classe *Coach*

C. O Sistema de Visão

As classes *FRNTVision* e *OMNIVision* foram desenvolvidas para responder à necessidade de comunicação entre os processos da visão e o processo de controlo.

A visão possui dois processos independentes que correspondem às duas câmaras existentes em cada robô: frontal e omnidireccional. A informação reunida por cada uma sobre os objectos ao seu alcance é escrita na classe correspondente; *FRNTVision* ou *OMNIVision*.

Apresentam-se em seguida as hierarquias das classes *FRNTVision* (Figura 11) e *OMNIVision* (Figura 12).

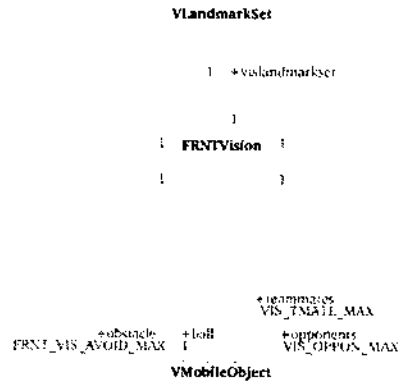


Figure 11 - Classe *FRNTVision*

Como se pode observar, estas classes são compostas por várias classes que modelam objectos físicos existentes no campo. A classe *FRNTVision* armazena informação relativa a objectos passíveis de serem detectados na câmara frontal (bola, obstáculos, adversários, colegas de equipa e referências do campo como balizas ou postes). De forma semelhante, a classe *OMNIVision* armazena informação relativa a objectos passíveis de serem detectados na câmara omnidireccional tais como: bola, obstáculos, linhas do campo, etc. A informação contida em ambas as classes é posteriormente integrada (pelo processo de controlo) no "estado do mundo".

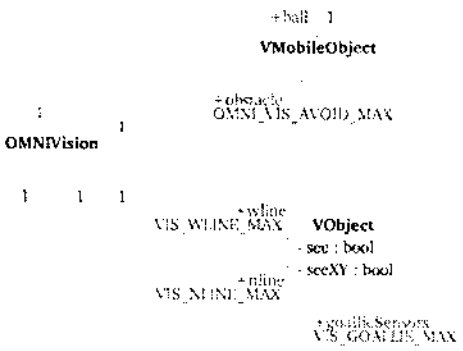


Figure 12 - Classe *OMNIVision*

As classes *FRNTVision* e *OMNIVision* usam as classes do tipo *VLandmark* e *VMobileObject* (Figura 13).

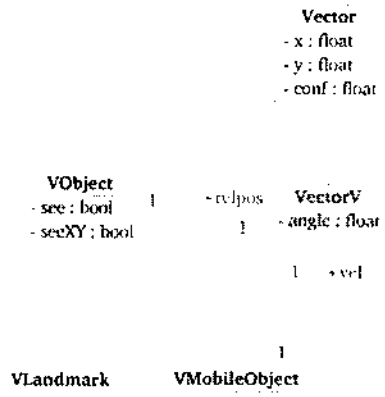


Figure 13 - Classe *VObject* e suas derivadas

Assim *VMobileObject* apenas difere de *VLandmark* por possuir uma velocidade, ou seja, objectos da classe *VMobileObject* são móveis.

Na classe *Vector* e na sua derivada *VectorV*, usada pela generalidade das classes associadas à interface com o sistema de visão artificial, os atributos x e y armazenam as coordenadas cartesianas do objecto visto na área corrigida da imagem. Se o objecto for visto fora da área corrigida, o sistema de visão fornece apenas o ângulo relativo a que o mesmo se encontra. Em ambos os casos existe uma confiança associada à medida.

Finalmente, a classe *VLandmarkSet* acomoda todas as *landmarks* (exceptuando as linhas) que a câmara frontal pode detectar (ou seja, balizas e postes).

C.1 *Os parâmetros das Câmaras* - Camera Settings

Verificou-se que cada câmara possui parâmetros de calibração específicos. Para armazenar estes pârametros desenvolveu-se a classe *CameraSettings* ilustrada na Figura 15.

Como se pode observar, esta classe possui atributos que permitem guardar toda a informação que caracteriza a

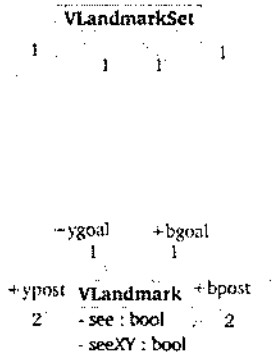


Figure 14 - Classe *VLandmarkSet*

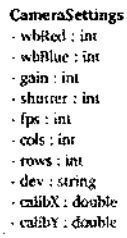


Figure 15 - Classe *CameraSettings*

calibração de uma câmara (balanceamento de "brancos", ganho, geometria da imagem, etc.).

D. Os Pontos de Referência do Campo - Landmarks

Uma vez que um campo de futebol robótico possui um conjunto de pontos de referência estáticos que podem auxiliar os robôs a localizarem-se no terreno de jogo, desenvolveu-se uma classe que permite guardar a localização real dos mesmos, i.e., a sua posição real no campo e não a percepção do robô (Figura 16).

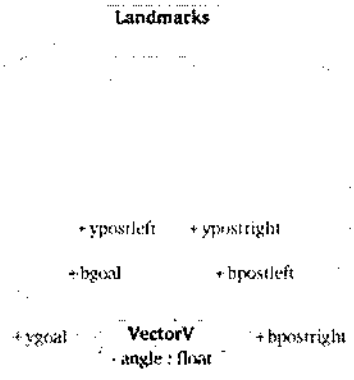


Figure 16 - Classe *Landmarks*

A informação recolhida pela visão em *FRNTVision* pode ser validada com auxílio da informação existente em *Landmarks*.

Convém notar que, embora a classe *VectorV* seja usada no âmbito da estrutura *Landmarks*, o atributo *angle* não é usado uma vez que se conhece com confiança máxima (1) a localização da marca no campo.

V. INTEGRAÇÃO DE INFORMAÇÃO

A integração de informação pretende construir um modelo interno do mundo tão preciso quanto possível, baseando-se nos dados sensoriais próprios e partilhados e nas acções executadas. Esta integração é realizada ao longo do tempo, minimizando o efeito do ruído normal dos sensores e a ausência momentânea de informação. Os dados que constam do modelo interno do mundo foram já referidos na secção anterior.

O posicionamento do robô no campo constitui o tema central desta secção. Como é facilmente perceptível, o conhecimento desse dado assume uma importância crucial numa equipa de futebol robótico. Os motivos que contribuem para a importância do conhecimento da posição absoluta descrevem-se em seguida:

- estimar as posições relativas das *landmarks* (balizas, postes, limites do campo, etc.);
- estimar as posições relativas dos seus *teammates* por forma a poder interagir com os mesmos (executar passes, etc.);
- aumentar o nível de coordenação da equipa, ao tornar possível a utilização de formações e tornar mais facilmente reconhecível qual a tarefa a desempenhar;
- permitir o planeamento de trajectórias;
- evitar colisões entre robôs e entre robôs e elementos do campo;
- decidir sobre qual a acção a executar.

O mecanismo de *Integração de Informação* foi desenvolvido tendo por base *know-how* proveniente da equipa "FC Portugal" [13]-[16] e de robótica móvel [17], [18].

As seguintes subsecções descrevem as limitações observadas na odometria e os mecanismos de correcção desenvolvidos para as ultrapassar.

A. Odometria

O sistema de odometria oferece tipicamente um mecanismo de localização simples, contudo, não suficientemente robusto para fornecer uma estimativa consistente da posição de um robô no decorrer de uma partida completa de futebol.

Foi levado a cabo um conjunto de experiências para avaliar o erro posicional proveniente do uso exclusivo de odometria na localização de um robô da equipa CAMBADA. Este conjunto de experiências foi realizado utilizando um robô que desempenha o papel de atacante até percorrer uma distância pré-definida. Durante a experiência um humano está constantemente a desviar a bola do robô obrigando-o a perseguir a bola. Ao fim de cada 10 metros são registadas a posição real e a posição indicada pela odometria. Em testes preliminares verificou-se que quando o robô é forçado a arranques e paragens súbitas e frequentes, o erro associado à odometria do robô aumenta muito rapidamente. Assim os testes foram efectuados de forma a evitar este tipo de arranques e paragens. Ao todo, foram realizados 10 percursos de

100 metros.

A.1 Erro em X e Y

As figuras 17 e 18 mostram a dependência do erro médio associado às componentes X e Y da posição absoluta em função da distância percorrida pelo robô. As barras verticais indicam o desvio padrão associado a cada valor médio de erro.

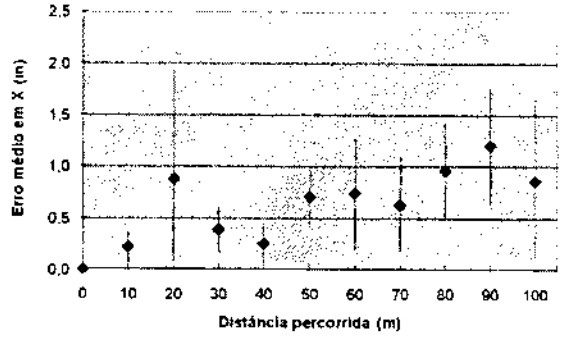


Figure 17 - Erro na componente X da posição absoluta

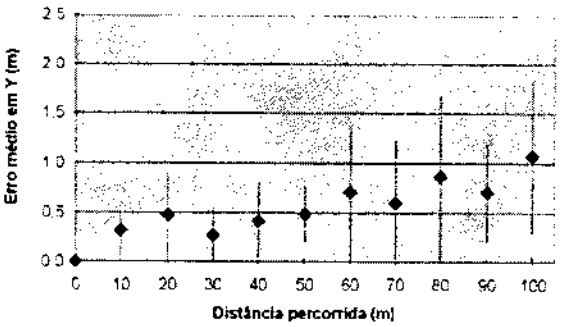


Figure 18 - Erro na componente Y da posição absoluta

Observa-se que, em média, o erro em ambas as componentes da posição absoluta tende a aumentar com a distância percorrida.

A.2 Erro posicional (módulo)

A Figura 19 mostra o valor médio do erro posicional (em módulo) e respectivo desvio padrão em função da distância percorrida. A figura inclui uma linha de tendência de evolução do valor do erro com a distância.

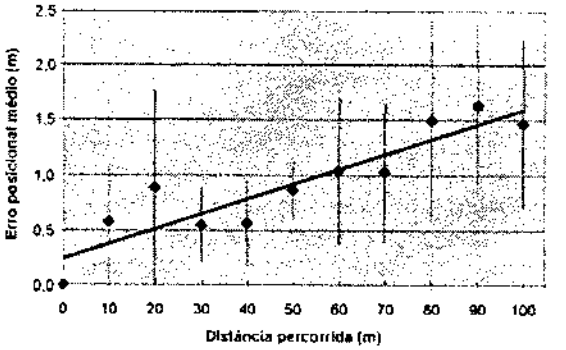


Figure 19 - Erro Posicional *versus* Distância

Conclui-se das experiências realizadas que, para distâncias percorridas até a $100m$, o erro posicional médio é uma função aproximadamente linear da distância percorrida, tendendo para cerca de 1.5% da distância percorrida. Verifica-se que para o conjunto dos 10 percursos realizados, e considerando todos os momentos de observação, o erro médio global situa-se em torno de 1 metro. Este valor pode ser considerado aceitável, dado que o campo de futebol tem habitualmente a dimensão de 12 metros de comprimento por 8 metros de largura.

Convém, no entanto, notar, como já foi referido, que estas experiências foram realizadas com o cuidado de evitar situações que reconhecidamente aumentam o erro de forma significativa. Além disso, mesmo considerando o pior caso dado pela figura 19 (média + desvio padrão), já se obtem níveis de erro menos bons ($\sim 1.6m$). Mais, ao aproximar-se dos 100 m, o pior caso do erro atinge cerca de 2.5 metros, valor que já se aproxima de uma situação de incerteza causadora de problemas ao nível, por exemplo, da coordenação da equipa.

Por estas razões, é necessário desenvolver mecanismos paralelos à odometria que forneçam uma melhor estimativa da posição do robô no decorrer de uma partida completa. Estes mecanismos são apresentados nas subsecções seguintes.

B. Calibração visual da posição absoluta do robô

Desenvolveram-se dois mecanismos distintos de calibração visual da posição absoluta. O primeiro mecanismo de calibração baseia-se no facto de que é possível realizar uma melhoria da estimativa da posição absoluta com base no conhecimento da posição relativa de uma *landmark* (na presente circunstância um poste). O segundo mecanismo de calibração é baseado no conhecimento da posição relativa de duas *landmarks* (dois postes) permitindo fazer a triangulação com o robô e determinar a sua posição absoluta com maior confiança. Nenhum dos mecanismos é totalmente infalível dado que ambos se baseiam em posições obtidas a partir do sistema de visão (que possui erros associados), no entanto, verificaram-se melhorias de desempenho assinaláveis com o seu uso. As secções seguintes abordam mais detalhadamente cada um dos mecanismos.

É importante notar que, tal como acontece com o sistema de odometria, também o sistema de visão fornece informação afectada por erros. Esses erros resultam de várias circunstâncias. À partida, as câmaras deverão estar posicionadas nos robôs exactamente na posição definida na fase de projecto. Para o garantir, é necessário calibrar a configuração física do sistema com alguma regularidade. Entretanto, mesmo nesse pressuposto, pequenas irregularidades no campo, ou até as próprias oscilações resultantes do movimento do robô, levam a alterações pontuais da posição das câmaras relativamente ao campo. Essas alterações podem levar o robô a ver objectos, não nas suas reais posições, mas sim a alguns (ou mesmo muitos) metros de distância. Na fig. 20, apresenta-se, para distâncias entre 0 e 15m, a curva do erro na distância dada pelo sistema de visão (em m), quando a inclinação da câmara frontal é superior à inclinação projectada por uma diferença de 1°. Como se

vê, o erro é tolerável apenas até 6m de distância. A partir dos 8m, a utilização da informação do sistema de visão exige grande cuidado. Esta é uma restrição importante dado que um campo tem normalmente 8 m por 12 m.

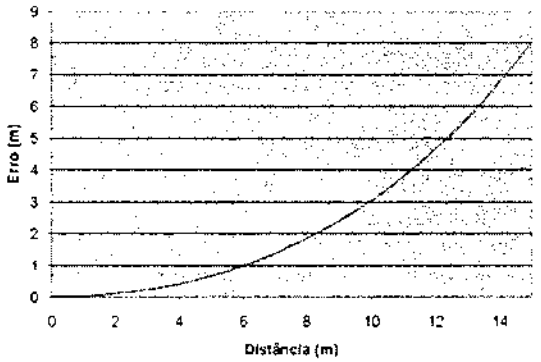


Figure 20 - Erro Posicional resultante de uma diferença de 1° na inclinação da câmara.

B.1 Calibração baseada num único poste

Como referido, o mecanismo de calibração baseado na posição de um único poste permite apenas realizar uma melhoria à estimativa da posição absoluta e não uma correcção completa do seu erro. Este facto pode ser observado na figura 21. Note-se que a direcção do robô é assinalada com uma seta.

Na posição 1, a frente do robô faz um ângulo ø com o poste e está a uma distância d do mesmo. Na posição 2 repetem-se as circunstâncias, ou seja, a frente do robô faz um ângulo ø com o poste e está a uma distância *d* do mesmo. Verifica-se claramente que não é possível calibrar a posição do robô com base em apenas um *landmark* (poste) dado que estando o robô em posições distintas poderá "ver" o poste na mesma posição relativa.

Assim, usando este mecanismo de calibração é apenas possível melhorar a estimativa da posição absoluta. A figura 22 ilustra o conceito. Este mecanismo de calibração parcial baseia-se em três considerações:

- Assume-se que erros na estimativa actual da posição absoluta do robô não são significativamente elevados e portanto esta pode ser usada no cálculo da nova estimativa.
- Com base na distância relativa ao poste, é possível traçar uma circunferência que indica as posições absolutas em que o robô poderia estar com base na distância relativa ao poste. Se se traçar uma recta entre o poste e a posição estimada[4] do robô intersecta-se a circunferência num ponto que seria o ponto de calibração. Contudo, este ponto não é usado uma vez que os erros associados ao sistema de visão levaram-nos a pesar com igual peso a estimativa existente e o valor da distância obtido através da visão.
- Cruzando a informação anterior (realizando uma média pesada entre as posições actual e induzida a partir da visão) obtém-se a estimativa final também assinalada na figura 22.

[4]Posição estimada com base na informação existente no "Estado do Mundo" do robô.

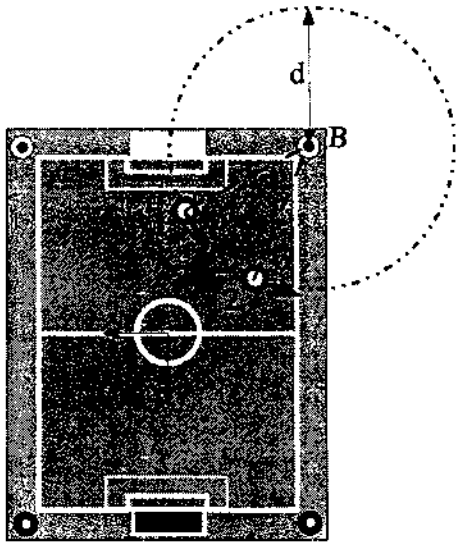


Figure 21 - Calibração posicional baseada num poste - o problema

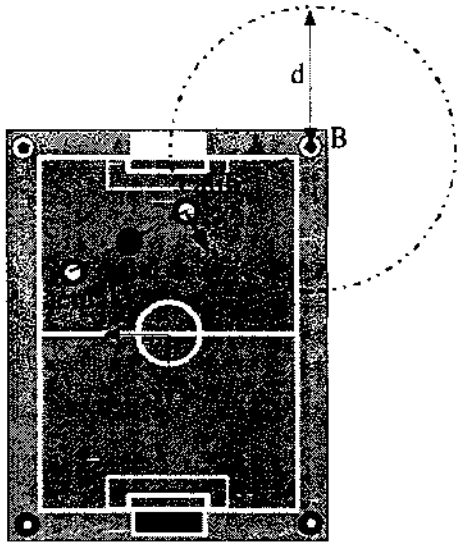


Figure 22 - Calibração posicional baseada num poste - a solução

B.2 Calibração baseada em triangulação

A correcção da posição absoluta baseada num único poste mostrou-se insuficiente para que o robô possuísse uma estimativa consistente da mesma. A solução descrita de seguida baseia-se no facto de que é possível corrigir integralmente a posição absoluta de um robô a partir de duas *landmarks* (nestas circunstâncias, dois postes).

A Figura 23 identifica um *set-up* de demonstração da teoria associada. Nesta figura, observa-se um triângulo constituído por dois postes e pelo robô. As letras maiúsculas referem-se aos ângulos e as minúsculas ao comprimento dos segmentos de recta. Considerou-se ainda um sistema de coordenadas cartesiano com as direcções assinaladas na figura.

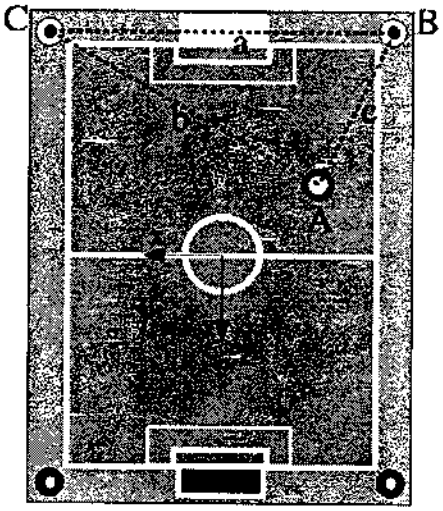


Figure 23 - Calibração posicional baseada em triangulação

A equação 1 permite determinar o ângulo entre o segmento de recta que une os dois postes e o segmento de recta que une o poste esquerdo ao robô (na figura 23 - ângulo C) a partir apenas das distâncias entre os três objectos.

$$C = \arccos\left(\frac{a^2 + b^2 - c^2}{2.a.b}\right) \quad (1)$$

A equação 2 (que varia conforme os eixos considerados) permite realizar a calibração da posição absoluta do robô com base no ângulo anteriormente determinado, nas distâncias entre os três elementos e na posição absoluta do poste esquerdo.

$$Robot(x,y) = \Big(Ypostleft_x - b.\cos C, \\ Ypostleft_y + b.\sin C\Big) \quad (2)$$

Verificou-se experimentalmente que apenas em ocasiões muito raras o robô "vê" os dois postes em simultâneo. Tal deve-se sobretudo à baixa resolução das imagens do sistema de visão e ao facto de que este sistema não possui visão omnidireccional a mais do que 1m.

Para ultrapassar esta limitação, desenvolveu-se um mecanismo de correcção activa da posição absoluta. Este mecanismo caracteriza-se por um comportamento especial através do qual o robô calibra a sua posição, mas que implica o abandono do seu envolvimento directo no jogo durante algum tempo. Em seguida descreve-se sucintamente este comportamento.

1. Rotação do robô de 360 graus pesquisando a existência de postes. Para cada poste detectado armazena-se o ângulo e a distância correspondentes.
2. Após a conclusão da rotação verifica-se se foram encontrados 2 ou mais postes:
   (a) Se não – o procedimento de calibração termina sem a realizar.
   (b) Se sim – os dados recolhidos são filtrados de forma a eliminar possíveis inconsistências. Após esta filtragem verifica se existem postes em posições consistentes e em número suficiente (2 no mesmo meio-campo) para realizar uma calibração completa:
      i. Se não – o mecanismo de correcção termina sem a realizar.
      ii. Se sim – usa as equações apresentadas anteriormente para realizar uma calibração completa da posição absoluta do robô.

Como se pode observar pela descrição anterior, nesta fase não se considerou a possibilidade dos postes pertencerem a lados distintos do campo. Optou-se por esta solução dado que quando o robô está de um dos lados do campo apenas consegue detectar os postes existentes na sua proximidade, logo, os postes que se encontram do lado oposto do campo dificilmente serão visíveis, e se o forem, o erro associado às suas distâncias faz com que sejam "eliminados" no processo de filtragem descrito.

C. *Outros mecanismos de tratamento de erros*

Para além dos mecanismos de calibração visual da posição absoluta descritos acima, outros mecanismos de tratamento de erros foram desenvolvidos, o quais se passa a descrever.

C.1 *Calibração da rotação*

Em alguns testes controlados (apenas rotação, apenas translação, etc.), verificou-se que o sistema de odometria fornecia valores de rotação afectados de erros sistemáticos, os quais se revelaram difíceis de anular na camada de *software* de baixo nível. Este problema acaba também por se reflectir nas componentes X e Y da posição absoluta.

Para corrigir este erro, mede-se o incremento de rotação em cada iteração do ciclo de controlo, multiplicando-o por um factor de calibração. Tipicamente existem dois factores de calibração de rotação, um para o movimento no sentido horário dos ponteiros do relógio e outro para o sentido oposto. É ainda de salientar que os parâmetros de calibração são específicos de cada robô dado que cada um deles possui uma geometria física própria.

Os dados experimentais expostos anteriormente foram colhidos tendo já activado este mecanismo de calibração.

C.2 *Tratamento de inconsistências*

Numa situação de funcionamento normal do robô, admite-se que a estimativa da posição do robô é suficientemente fiável, podendo-se derivar as posições relativas dos restantes elementos fixos existentes no terreno de jogo. Quando um determinado elemento imóvel (poste, baliza, etc.) é detectado pelo sistema de visão e transmitido ao sistema de controlo de alto nível, este verifica se o mesmo se encontra numa posição coerente com o seu "estado do mundo". Isto significa que, se a posição do elemento não se aproximar (por uma determinada margem) da estimativa actual da sua posição, a informação recém-adquirida é ignorada, sendo então incrementado um contador de inconsistências. Quando o número de inconsistências atinge um determinado limite[5], o processo de controlo instrui o

[5]O facto de existir um número elevado de inconsistências é indicativo de que a posição absoluta do robô está corrompida. Nestas circunstâncias é imperativo que se realize uma calibração da posição absoluta.

robô para realizar uma calibração activa (ver subsecção *Calibração baseada em triangulação*).

C.3 Limites nas coordenadas X e Y

Quando o robô se aproxima dos limites físicos do campo, um erro posicional, ainda que pequeno, facilmente resulta em coordenadas *X* e *Y* absolutas que excedem esses limites. Esta situação constitui uma oportunidade de calibração. Assim, sempre que uma das coordenadas sai do intervalo de valores possíveis (num campo normal, -7.0m a 7.0m em Y e -5.0m a 5.0m em X), o respectivo valor é corrigido no Estado do Mundo, passando a ser o limite que foi ultrapassado.

C.4 Integração de informação ao longo do tempo

Com vista à construção de um Estado do Mundo mais fiável do que aquele que se obtém através de calibrações pontuais, foi iniciado o desenvolvimento de um módulo de integração de informação ao longo do tempo. Neste caso, em vez de usar informações pontuais sobre a posição dos pontos de referência (objectos fixos como balizas, postes, etc.), mantém-se internamente actualizada a posição relativa de cada um deles. De cada vez que um dado ponto de referência é detectado, a respectiva posição relativa é actualizada através de uma média ponderada com a posição actual, desde que não haja inconsistência clara entre ambas. Entretanto, dado que os pontos de referência têm uma posição relativa (de uns ao outros) imutável, os valores das posições desses pontos de referência relativamente ao robô são corrigidas por forma a aproximarem-se da sua posição correcta no campo.

Este módulo de integração de informação sobre os pontos de referência ainda não está devidamente avaliado.

VI. CONCLUSÃO

Este artigo apresentou a arquitectura de software de um agente robótico da equipa CAMBADA focando essencialmente nas estruturas de dados e nos mecanismos de integração de informação. Para maior clareza, as estruturas de dados são apresentadas em UML, sendo também identificados os problemas associados.

Os mecanismos de integração de informação que foram desenvolvidos e descritos têm por objectivo principal estimar as posições absolutas do robô e dos outros objectos móveis existentes no campo. Nesta fase, integra-se essencialmente informação odométrica e visual. Na continuação dos trabalhos, estão a merecer atenção os seguintes aspectos:

- Desenvolver mecanismos adicionais de calibração da posição do robô.
- Avaliação e eventual utilização do mecanismo de integração de informação ao longo do tempo.
- Desenho e implementação do sistema de gestão de confianças.
- Integração de informação ao nível da equipa.

REFERENCES

[1] CAMBADA Homepage, http://www.ieeta.pt/atri/cambada/, February, 2005

[2] RoboCup Official Web Site, http://www.robocup.org/, February, 2005

[3] Transverse Activity on Intelligent Robotics Homepage, http://www.ieeta.pt/atri/, February, 2005

[4] Institute of Electronics and Telematics Engineering of Aveiro Homepage, http://www.ieeta.pt/, February, 2005

[5] Fédération Internationale de Football Association, http://www.fifa.com/en/index.html, February, 2005

[6] RoboCup, Middle Size Robot League Rules and Regulations for 2004, Draft Version pre-8.3, June, 2004

[7] Wi-Fi Alliance, http://www.wi-fi.org, February, 2005

[8] L. Almeida, F. Santos, T. Facchinetti, P. Pedreiras, V. Silva and L. Seabra Lopes, "Coordinating Distributed Autonomous Agents with a Real-Time Database: The CAMBADA Project", Computer and Information Sciences - ISCIS 2004: 19th International Symposium, Proceedings, Aykanat, Cevdet; Dayar, Tugrul; Korpeoglu, Ibrahim (Eds.), Lecture Notes in Computer Science, Vol. 3280, p. 876-886.

[9] F. Santos, L. Almeida, P. Pedreiras, L. Seabra Lopes, T. Facchinetti, "An Adaptive TDMA Protocol for Soft Real-Time Wireless Communication among Mobile Autonomous Agents", Proc. WACERTS'2004 (em publicação).

[10] Unified Modeling Language Resource Page, http://www.uml.org/, February, 2005

[11] James Rumbaugh, Ivar Jacobson, Grady Booch, "The Unified Modeling Language Reference Manual", Addison-Wesley, 1999

[12] Martin Fowler, Kendall Scott, "UML Distilled Second Edition - A Brief Guide to the Standard Object Modeling Language", Addison-Wesley, 2000

[13] FC Portugal Homepage, http://www.ieeta.pt/robocup/index.htm, February, 2005

[14] L.P. Reis, J.N. Lau, E.C. Oliveira, "Situation Based Strategic Positioning for Coordinating a Team of Homogeneous Agents", Balancing Reactivity and Social Deliberation in Multi-Agent Systems, Markus Hannebauer, Jan Wendler, Enrico Pagello, editors, LNCS 2103, 175-197, Springer Verlag, 2001

[15] L.P. Reis, J.N. Lau, "FC Portugal Team Description: RoboCup 2000 Simulation League Champion", RoboCup-2000: Robot Soccer World Cup IV, Peter Stone, Tucker Balch and Gerhard Kraetzschmar editors, LNAI 2019, 29-40, Springer Verlag, Berlin, 2001

[16] J.N. Lau, L.P. Reis, "FC Portugal 2001 Team Description: Flexible Teamwork and Configurable Strategy", RoboCup-2001: Robot Soccer World Cup V, Andreas Birk, Silvia Coradeshi, Satoshi Tadokoro editors, LNAI, Springer Verlag, Berlin, 2002

[17] L. Seabra Lopes, J.N. Lau, L.P. Reis, "Intelligent Control and Decision-Making demonstrated on a Simple Compass-Guided Robot", Proceedings of SMC'2000, IEEE Int. Conference on Systems, Man and Cybernetics, Nashville, USA, October 2000

[18] L. Seabra Lopes, "Carl: from Situated Activity to Language Level Interaction and Learning", Proc. IEEE Int'l Conf. on Intelligent Robots and Systems (IROS), Lausanne, Switzerland, p. 890-896, 2002

Architecture and basic skills of the FC Portugal 3D simulation team

Hugo Marques, Nuno Lau, Luis Paulo Reis

Resumo -**A liga de simulação do RoboCup, iniciou em 2004 uma nova competição que utiliza um simulador de futebol em que o ambiente virtual tem 3 dimensões. Este artigo descreve a arquitectura dos agentes da equipa FC Portugal 3D que concorreram ao campeonato mundial do RoboCup 2004 em Lisboa. Serão descritas as características do novo simulador 3D e os aspectos principais da arquitectura da equipa FC Portugal e do desenvolvimento dos comportamentos básicos dos agentes. A equipa FC Portugal classificou-se em 8º lugar da competição 3D do campeonato mundial de 2004 do RoboCup.**

Abstract – **The RoboCup Simulation League introduced in 2004 a new competition based on a soccer simulator which implements a 3D virtual environment. This paper presents the architecture of the agents made by the FC Portugal 3D team in order to participate in the RoboCup 2004 World Championship competition which happened in Lisbon. We will describe the new 3D simulator and the most important characteristics of the architecture, basic behaviour and skills of the agents developed agents. The FC Portugal 3D team achieved 8th place in the RoboCup 2004 World Championship.**

I. INTRODUCTION

The first version of the 3D simulation league simulator was made available to the RoboCup community during January 2004. The proposal of a new simulator had the following objectives:

- Replace the 2D environment of previous simulator with a 3D environment
- New, more realistic, physics model
- Simulation results should not be dependent on available computational power or on the quality of network resources

The first version of the simulator was very immature. Still it allowed us to contact with some of the new models of robots, of their sensors and actuators, and some of the new features related with the physics model and synchronization of agents with simulator.

The differences between the new 3D simulator [1] and the 2D simulator [2] used in previous RoboCup competitions, and in our previous research, are very significant. These differences led us to the decision of starting to develop from scratch a new agent for the new 3D simulator. Of course, we intended to apply, with proper adaptations, most of the methodologies we had previously developed for the 2D simulator [3-5]. However the code of the new agent is almost completely new and did not result from the adaptation of our 2D agent code.

The following sections describe the architecture and some of the algorithms that are used by the new FC Portugal 3D agent.

II. SIMULATION ENVIRONMENT FOR 3D SOCCER

The simulation environment of the RoboCup 3D Simulation League is based on a client-server model. The simulator is the server and agents and visualization tools are the clients. The simulator creates the virtual environment (soccer field, markers, goals, etc...) where agents live, sends sensory information to the agents, receives their actions and applies the virtual physics model in order to resolve positions, collisions and interactions with the ball. Each team plays with 11 agents that must cooperate to score as much goals as possible while not allowing the other team to score.

The development of the 3D simulator used available open-source tools extensively. It uses the SPADES [6,7] framework for the management of agent-world communication and synchronization, ODE [8] for the physical model, expat [9] for XML processing, Ruby [10] for scripting language support and boost [11] for several utilities.

A. SPADES

The 3D simulation server is implemented above a platform called SPADES (System for Parallel Agent Discrete Agent Simulation) [6]. SPADES is a middleware system for agent-based distributed simulation. It aims to provide a generic platform to run in multi-computer systems. It implements the basic structure to allow the interaction between agents and a simulated world so that the users do not have to worry about communication and synchronization mechanisms such as sockets, addresses, etc.

SPADES' main features are:

- Agent based execution - support to implement sensations, thinking and actions.
- Distributed processing – support to run the agents applications on many computers.
- Results unaffected by network delays or load variations among the machines – SPADES ensure

that the events are processed in the appropriate order.

- Agents can be programmed independently from the programming language – the agents can be programmed in any language once it provides methods to write/read to/from Pipes.
- Actions do not need to be synchronized in the domain – the actions of the agents can take effect at varying times during the simulation.

A.1 Components organization

SPADES components are organized in a client-server architecture (Fig. 1). The Simulation Engine and the Communication Server are provided by SPADES; while the Agents and the World Model are built by the user and run upon the formers.

The Simulation Engine is a generic piece of software that provides abstractions to create specific world models upon it. Agents may run in the same computer of the Simulation Engine on in remote computers linked to the network, in this case a Communication Server must be running in the remote computer. The World Model module must be running in the same computer of the Simulation Engine. This module specifies the characteristics of the environment where the agent will live.

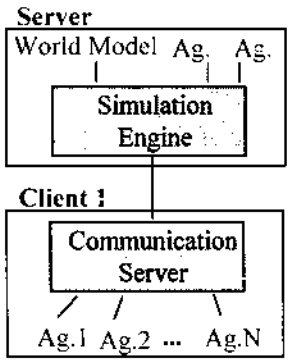


Fig. 1 – SPADES Components diagram.

A.2 Sense-Think-Act Cycle

SPADES implements what it calls the sense-think-act cycle in which each agent receives sensations and replies with actions. That means that an agent is only able to react after receiving a sensation message. The agent is also capable of requesting its own sensations, but the principle remains - a sensation must always precede an action. In order to allow actions between "normal" sensations, SPADES provides an action called *request time notify* that returns an empty sensation and after receiving it the agent is able to respond with actions. For example, if an agent received a sensation at cycle 100 and wants to produce an action at cycle 110, and if the next sensation will only arrive at cycle 120, the agent can ask to receive a *time notify message* at cycle 110 and just reply with the desired action after receiving it.

Fig. 2 depicts the sense-think-act cycle and the time where each of its components run. From A to B a sensation is sent to the agent. After receiving the sensation (from B to C) the agent decides which actions will be executed; then from (C to D) the actions are sent do the server.

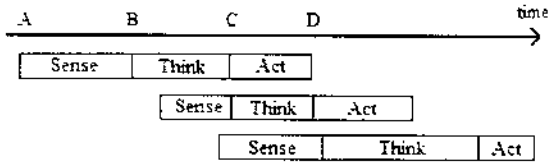


Fig. 2 – SPADES Sense-Think-Act Cycle.

In many agents, the sense, think and act components may be overlapped in time (like in Fig. 2). There is just one restriction: The thinking cycles for one agent cannot be overlapped. This constraint makes sense, since just a single processing unit is used per agent, and thus, just one sensation at time can be processed.

B. SIMULATION SERVER

As stated before the simulator runs upon the SPADES, and uses ODE to calculate the physical interactions between the objects of the world. The graphical interface is implemented using OpenGL.

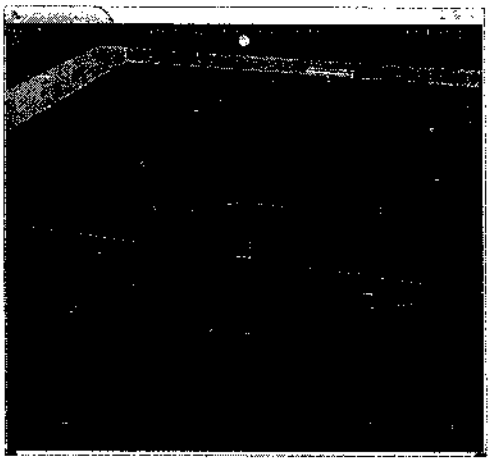


Fig. 3 – Snapshot of the 3D Simulator.

The 3D simulation server (Fig. 3) [1] allows twenty two agents (eleven from each team) to interact with the server in order to play a simulated robotic soccer game. Each agent receives sensations about the relative position of the other players and field goals and other information concerned with the game state and conditions. At this time the information about the positioning of the objects in the world is given by an awkward omnivision that allows the agent to receive visual information in 360 degrees. The agents have the shape of a sphere. Replying each sensation an agent sends actions like *drive* or *kick*. Driving implies applying a force on the body with a given direction and kicking implies applying a force on the ball radially to the agent. Each sensation is received on every 20 cycles of the server and each cycle takes 10 ms.

B.1 Sensations

There are already several sensations that an agent is able to receive. Every sensation starts with the character 'S' followed by two integers (*time*). The first is the cycle in which the message was sent and the second is the cycle in which the message arrived to the agent. A sensation message has the format:

```
Stime time data
```

where *data* is the string with the information related with the sensation itself[1].

Vision. The vision sensation gives the agent the spatial arrangement of the world objects in the field. World objects are the players, the ball, the goals and flags in the corners. The vision is, on the 0.3d version, omni-directional and the objects are considered transparent (at least for the vision sense). The position of the objects is given in polar coordinates relative to the respective agent. The coordinates are given by the distance, the horizontal angle – theta – and the elevation angle – phi.

```
Stime time (Vision
  (Flag  (id id) (pol d theta phi)) …
  (Goal (id id) (pol d theta phi)) …
  (Ball (pol d theta phi))
  (teamname (id id) (pol d theta phi)) …
          )
```

GameState. The game state sensation gives the agent all the information concerned with the game properties. It gives information about the dimensions of the field, the goals, the ball and the agents. It also gives information about the mass of the objects in the world and other aspects of the game like: time, play mode agent number, and if the agent's team is the right or the left one. Here is the format of the given information:

```
Stime time (GameState
             (team side)
             (unum number)
             (FieldLength length)
             (BallMass mass)
             (playmode playmode)
              …)
```

AgentState. The agent state gives the agent information about its internal state. For now, just information regarding the battery condition and the temperature are given.

```
Stime time (AgentState
             (battery battery)
             (temp temp)
            )
```

[1] Note that one sensation message can have more than one sensation.

B.2 Actions

As well as sensations, several actions are implemented that allow an agent to interact with the world. Every action message starts with the 'A' character and it is followed by a string,

```
Adata
```

where *data* contains the information about the action itself.

Create. The first action that an agent must send is the create action. This action allows the server to register an agent and thus establish the communication with it. The action create as the form:

```
A(create)
```

Init. The init message allows the server to receive essential information about the agent, namely its number and its team. If the number is passed as 0, the server automatically attributes a number to the agent. The init action has the format:

```
A(init (unum number) (teamname name))
```

Beam. The beam action allows an agent to move to a given point. It may only be executed before game kickoff and it does not obey to any physical law. Its structure is the following:

```
A(beam x y z)
```

Drive. The drive action allows an agent to move. It applies a force vector (x, y, z) to the center of the agent's body with the maximum length of 100.0 units. It has the format:

```
A(drive x y z)
```

Kick. The kick action follows the laws of gravity and movement of physics and allows an agent to kick the ball with a given force intensity and a given vertical direction (see *3.4 Physics – Kick*). The vertical direction is passed as an argument - the elevation angle. The horizontal direction is radial to the agent's body. The action must be sent like:

```
A(kick angle force)
```

B.3 Other important communication procedures

The server establishes the communication by sending a *done* ('D') message (Riley *et al.* 2003). When the agent receives this message it should execute its initialization procedures and when it finishes them it must send an *initdone* ('I') message. After that the server starts to send

sensations and the agent replying with actions. Every set of actions must finish with a *done* ('D') message (Fig. 4).

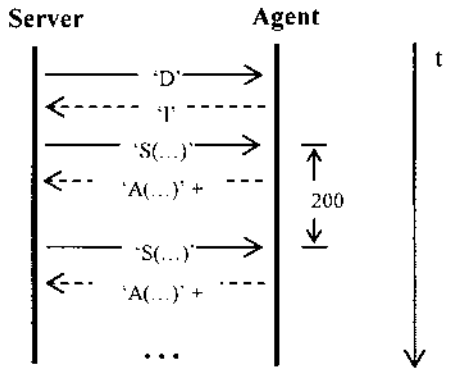


Fig. 4 – Temporal diagram of the communication between the agent and the server.

A sensation is received in every 20 server cycles, which means that an agent should only be able to execute actions within 20 cycles intervals. However, as stated before, an agent can ask the server to receive a sensation in a given cycle by sending a *request time notify* (R) message. The format of the message is the following:

```
Rtime
```

where *time* is the time at which the server must reply. This procedure makes the server reply with an empty sensation ('T') at the cycle given. As stated before the only reason to receive an empty sensation is to be able to act in a given time between two sensations.

An example where the *request time notify* makes sense is when an agent wants to kick the ball in given position. In the 3D server simulator, to kick the ball in a given direction, an agent must place itself quite accurately. Thus, because the interval between sensations is too long (200ms), it can happen that an agent that is running to the kicking point at cycle t is before that point and at cycle $t + 1$ has already passed the point. To surpass this, one can predict the time that the agent arrives to the desired position and ask to receive a time notify message at that time in order to be able to kick at the right moment.

By receiving this sensation the agent is able to respond with action messages.

Each sensation takes 10 cycles to reach the agent (send delay) and actions sent by the agent take 10 cycles to reach the simulator. Hence a sent action starts to take effect at the time the next sensation is sent by the server as it is shown in Fig. 5.

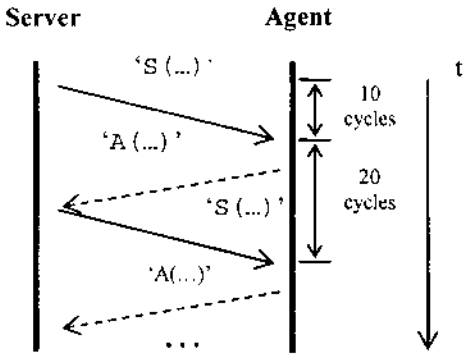


Fig. 5 – Communication delays.

B.4 Physics

The physical interactions of the game are made in a discrete way that is, in every cycle the new forces to be applied to the bodies, their current positions, velocities, etc. are calculated. Every cycle is simulated to take approximately 10 ms

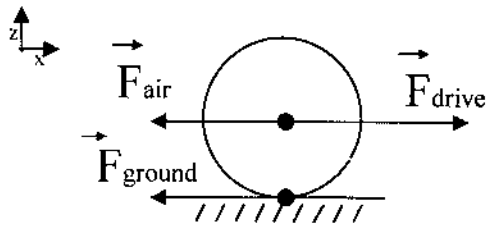


Fig. 6 – Forces applied to the body of an agent when the drive effector is used

During a regular drive, the body of an agent is under the influence of three forces: the drive force, the drag force caused by the contact of its body with the ground and the force caused by the friction of its body with the air. Additionally a drag torque is also applied. The ground force is such that the robot rotates without sliding over the field. The drag force and drag torque are proportional to the robot's speed. The drive force is controlled by the agent through the *drive* action.

III. FCPORTUGAL 3D AGENT

The FC Portugal 3D was developed from scratch. Any attempt to use the code of the FC Portugal 2D agents would run into serious problems, not because of the addition of an extra coordinate, but because of the huge difference in the 2D and 3D servers functioning.

A. Agent's Architecture

The agent structure includes six main modules/packages that cover different parts of its functioning (Fig. 7).

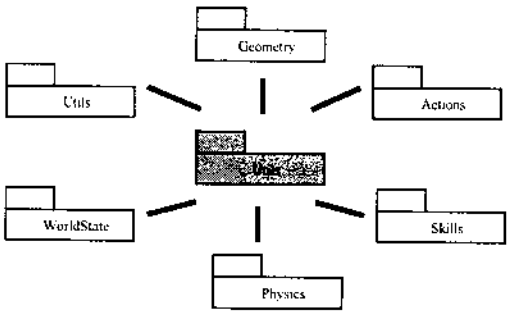


Fig. 7 – Modular division of FC Portugal 3D Basic Agent.

World State. The "World State" package (Fig. 8) is probably the most complex one. It has all the information that the FCP Agent needs to decide which action it should take. There are three kinds of information that the WorldState needs: information about the objects (like players, landmarks and the ball), information about the conditions of the game (like field length, goal width, etc.) and the state of the game (like the current play mode, the result, the time, etc).

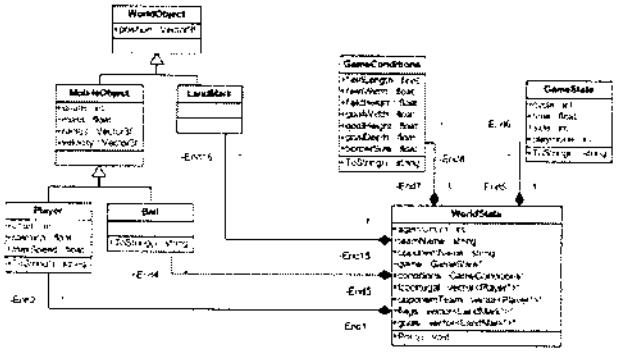


Fig. 8 – FC Portugal World State package architecture.

Physics The physics package aims to reproduce the physical interactions between the bodies in the world using the same model as the server does. At the present one can estimate the velocity and the acceleration of an object, the current forces applied in a given body, the breaking distance and the time that an agent takes to move from one position to another applying a given force.

Geometry. The "Geometry" module is used to make easier the execution of geometrical calculations. It is used to compute data concerning distances, vectors, etc.

Skills. The skills are the low-level actions that an agent is able to perform. Kicking the ball, moving its body, intercepting the ball, or dribbling are examples of agent's skills. These are also the ones implemented at the moment by FC Portugal team. The scheme of the skills architecture is in Fig. 9.

Every skill implements the *GenericSkill* interface. When a skill is initialized it immediately computes the necessary calculations to execute itself. However, the initialization does not execute the skill. Every skill has a method named *Execute()* that allows its execution.

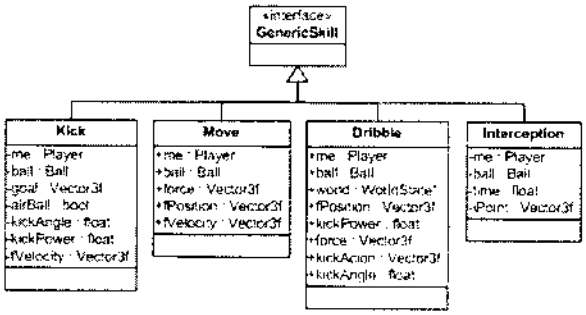


Fig. 9 – Skills' architecture

Actions. An action is a group of skills that, together, produce higher lever behaviours. Sample of actions may be: passing, shooting, making forwards, etc. Not all of the implemented actions are used in the current version of the FC Portugal team. However, the architecture of the FCP Agent supports the implementation of passes, shoots and forwards.

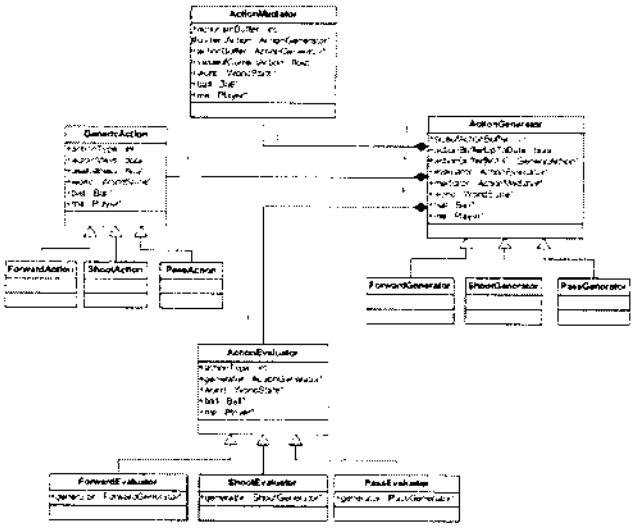


Fig. 10 – Actions Architecture.

In order to determine which action should be executed, four classes are involved: a mediator, an evaluator, a generator and the action itself. The generator (*ActionGenerator*) is the class that allows the creation of potential actions that are able be performed. There are 3 classes that extend the *ActionGenerator*, one per type of action – pass, shoot and forward. Each class is able to return a set of actions of its type, adjusted to the current situation, and that should be considered for future evaluation. The actions returned by each generator have their own properties according with its type and all of them extend the *GenericAction* class. The evaluation of the actions is done by the evaluator (*ActionEvaluator*). This is a class that enables the agent to estimate the usefulness of every generated action. The evaluator has also 3 classes (one per type of action) which extend the *ActionEvaluator* class; each of them has its one evaluation components that allow them to estimate the usefulness of every action of its type.

To join everything together the FCP Agent has a mediator (*ActionMediator*) which is a class that is able to call the necessary functions to generate and evaluate every type of action and to decide which will be the next action to be performed.

Utils. The package *Utils* was made to contain classes that do not have a direct relevance on the agent behaviour but help to make some tasks easier. Examples of the operations of these classes are the creation of log files, communication with the simulator, a message parser and a message composer to send the actions to the simulator.

B. Localisation system

The agent gets the objects position by the vision perceptor, which gives the relative position of all objects in the world in polar coordinates. The absolute position of the landmarks is set at the time the agent receives the field dimensions and the goals position. This information is usually sent by the server in the second or third sensation.

The localization algorithm is quite straightforward. The agent starts to seek the closer landmark. If that landmark is closer than 20.0m the agent determines its position by the absolute position of that landmark and its relative position to the agent. If the landmark is farther than 20.0m the agent combines, using a simple average, the position of the closer landmark with the position of the second closer one to determine its position.

Several experiments were conducted by moving the agent inside the pitch and determining its position using the algorithm described above. During these experiments the maximum error of the algorithm was around 20.0cm and the frequency of errors of this magnitude was very low.

C. Physics

The agent should be able to predict the future state of the world if he decides to act in a certain way. This knowledge is essential for making the right decisions. In order to accomplish this functionality, and as there is no documentation on these matters for the 3D simulator, several experiments were conducted to infer which is the physics model of the simulator. The results of these experiments are presented in this section.

To be able to capture the effect of a given driving force the agent must know the magnitude of the friction force and its own velocity. Two distinct methods were used to obtain these informations.

To calculate the force in the agent caused by the friction with the air quadratic regression was used. It is given by:

$$\vec{Fair} = A * \vec{v}^2 + B * \vec{v} + C$$

where A = -0.84, B = 179.9 and C = 112.3.

The graphic in Fig. 11 shows the performance of the formula used during a two step movement – acceleration using maximum force in the positive direction of the x axis followed by an acceleration also using maximum force in the opposite direction. One can see that in the first part of the movement (accelerating on the direction of the x axis) the approximation used is very good since the prediction of the agent is very near the server data. However, the second part, immediately after the agent starts to brake, one can see that there is a big difference between the server data and the agent's approximation, meaning that the agent's calculation is clearly not good enough.

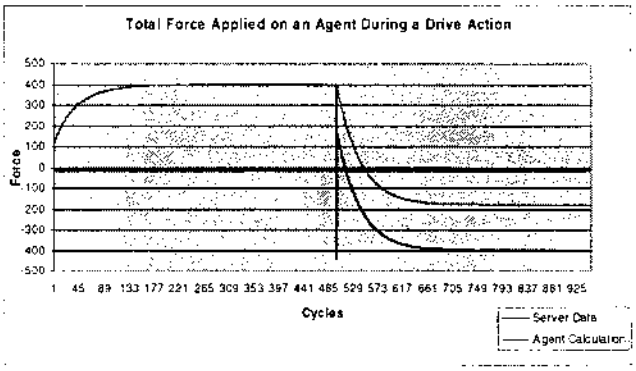


Fig. 11 – Graphic with the forces applied in the agent's body during a drive action

The player velocity was estimated based on the previous three positions of the agent assuming that acceleration was constant during that period. Analytically this provided the following equation for the velocity and acceleration:

$$\begin{cases} v_0 = \dfrac{4p_1 - 3p_0 - p_2}{2t} \\ a = \dfrac{p_2 - p_0 - 2v_0 t}{2t^2} \end{cases}$$

This methodology has been tested and Fig. 12 shows an example of the incurred errors.

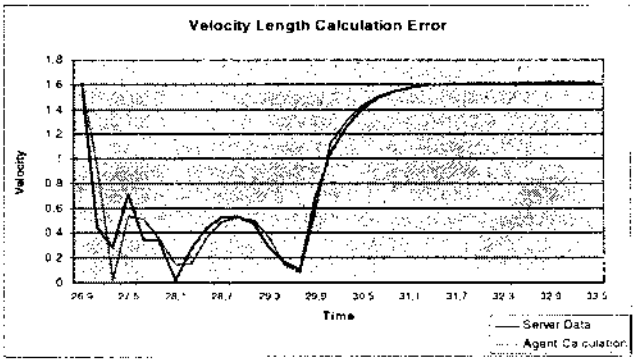


Fig. 12 – Error on the calculation of the length of the velocity vector between time 26.9s and 33.5s

To simulate the ball movement in the ground the laws of physics were applied to the situation of a free running ball with a drag force. This has led to the following velocity and position formulas:

$$v = e^{-\frac{K}{m^2}t} \times v_0$$

$$x = -2e^{-\frac{K}{m^2}t} \times v0 + 2 \times v0 + x0$$

Where x0 and v0 are the initial position and velocity of the ball. Fig. 13 shows that the results obtained are very close to the simulator's calculations. One cannot distinguish between the real values and the approximation made.

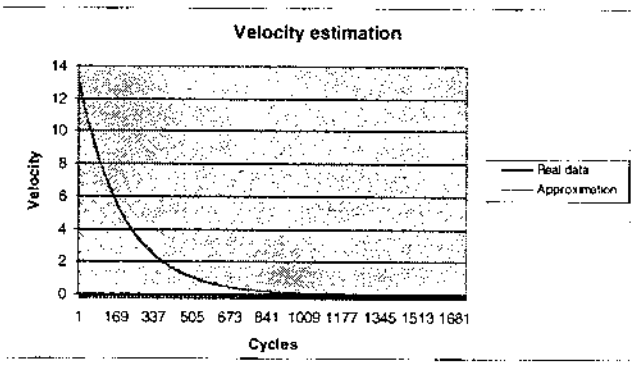


Fig. 13 – Graph showing the velocity difference between the server values and the approximation values given by the formula used

D. Skills

Move; The move skill takes three arguments: a pointer to the world object, a vector containing the point to where the agent wants to move and the velocity that it wants to arrive there. To calculate the force that should be applied at a given moment in order to move, the agent starts by getting the braking distance to reach a given velocity, given its own current position and velocity. Then it can estimate the distance that it can accelerate (*distanceToPoint – brakeDistance*). If that distance is made in more than forty cycles it accelerates, otherwise it brakes[2].

Kick; As already stated a kick action takes two arguments – the kick power and the kick angle. The FC Portugal kick skill receives four arguments: a pointer to the world object, a vector giving the point to where the ball should be kicked, a boolean informing if maximum power should be used and another boolean informing if the ball should be kicked by the air.

The power that an agent applies when kicking, is directly proportional to the distance to the final point of the kick plus an extra force. The extra force varies depending on whether the agent wants to kick the ball by air or not.

Dribble; Dribbling is the skill that allows an agent to run with the ball near its body. This is achieved by giving small kicks to the ball and running in order to catch it again. It starts by moving the agent to a position which enables it to kick the ball in the goal point (*finalPosition*) direction. Arriving there the agent tries to kick the ball as farther as possible (*maxDistance*). If there is an opponent agent that is able to steal the ball, the FCPagent starts to reduce the kick distance successively until none of the opponent players can catch the ball first.

Interception; The interception skill gives the agent the ability to catch the ball. At the moment just the quicker interception is calculated, that is, the interception that enables the agent to catch the ball in the less time possible. It receives, as arguments, a pointer to the world and the number and the team of the player to whom the interception will be calculated. The algorithm is the following: it calculates the position of the ball in each 20 cycles (interval between sensations), calculates the distance the player is able to run at maximum speed and the distance from the player to the ball. The agent is able to intercept the ball when the ball distance is smaller than the distance that is able to run in a given time.

E. Agent behaviour

```
Case (Playmode) equals
{
   BeforeKickOff:
      MoveAccordingSBSP();

   FCPortugalKickOff:
   {
      If (myNumber == 9)
         RunToTheBallAndKickIt();
      Else
         MoveAccordingSBSP();
   }

   OpponentKickOff or OpponentKickIn or
   OpponentCornerKick or OpponentGoalKick:
   {
      MoveAccordingSBSP();
   }

   FCPortugalKickIn or FCPortugalCornerKick
   or FCPortugalGoalKick or PlayOn:
   {
      If (MyMoveToTheBallAccordingSBSP())
         RunToTheBallAndKickIt();
      Else
         MoveAccordingSBSP();
   }
}
```

The agent behaviour is very much tied with the SBPS [3] originally developed by FC Portugal 2D team. The SBSP, was successfully used by FC Portugal team on the 2D simulator and consists in assigning a strategic position to each agent on the field given the position of the ball and the current situation. The player that runs to the ball is the one that has the best interception from its strategic position to the ball. Each agent is differentiated by its number [4].

Before the game starts each agent requests the position that it must occupy on the playing field given its number.

[2] 40 cycles = 20 cycles to start the action + 20 cycles to execute the action. The position where the agent will be at the time the action is executed must be estimated by the agent.

The SBSP has the advantage to make easier for an agent to know where its team mates are without having to calculate their real position. This is so, because each agent is able to know the position of its companions simply by running the SBSP algorithm for all the players of its team.

If the kick-off is assigned to the opposing team the FC Portugal team places itself according with the SBSP algorithm until the ball is touched by one of the opposing players. If the kick-off is assigned to the FC Portugal team, the agent with the closest distance from the position given by the SBSP to the ball, starts running to a position that allows it to kick the ball. Arriving there it kicks the ball. The other players position themselves according to SBSP until the former touches the ball.

After the ball is touched by one of the agents of the team which has the kick-off the game state is changed to "PlayOn" and the game starts. At this state each player moves to its strategic position by running the SBPS algorithm. The only exception is the player with the best interception time, this player tries to catch the ball and to kick it. The position to where the ball is kicked depends on where the ball is situated in the field. If the ball is near FC Portugal goal then the players try to kick it to the sides, in order to avoid putting it in a frontal area near the goal (which would be very dangerous in case an opponent was present). Otherwise FC Portugal agents try to shoot in the opponent's goal direction.

IV. CONCLUSIONS

The FC Portugal 3D team participated in RoboCup 2004 achieving the 8th place. It participated in 9 games having 6 defeats, 2 draws and 1 win. The low level skills of FCP agents performed differently on the competition computers and in the tests that had been previously performed. Although the team was able to reach the ball quite fast, most of the time the agent was not capable of kicking the ball at all, the time spent in positioning itself properly was so long that the opponents could arrive and steal the ball. Unfortunately, it was not possible to perform tests in the competition computers and because the problem did not occur in the computers used in our tests the source of the problem could not be found in effective time.

V. FUTURE WORK

A new world state update and move low-level skills must be developed so that FC Portugal agents can kick the ball confidently. Without this functionality it is impossible to play soccer efficiently. From this point on the FC Portugal team wishes to implement more ideas, which have proved to be successful on the 2D Simulation League, in the 3D Simulation League, namely, the SBSP, the Low Level Skills evaluators, and others like the use of intelligent communication.

ACKNOWLEDGEMENTS

This research is supported by FCT-POSI/ROBO/43910/2002 Project – "FC Portugal New Coordination Methodologies applied to the Simulation League".

REFERENCES

[1] RoboCup Soccer Server 3D Maintenance Group, "The RoboCup 3D Soccer Simulator" http://sserver.sourceforge.net/ 2003

[2] Chen, Mao et al. "RoboCup Soccer Server", http://sserver.sourceforge.net/, 2003

[3] L.P. Reis, N. Lau, E.C. Oliveira "Situation Based Strategic Positioning for Coordinating a Team of Homogeneous Agents", In: Balancing Reactivity and Social Deliberation in Multi-Agent Systems, Markus Hannebauer, Jan Wendler, Enrico Pagello, editors, LNCS 2103, pp. 175-197, Springer Verlag, Berlin, 2001

[4] L.P. Reis and N. Lau "FC Portugal Team Description: RoboCup 2000 Simulation League Champion". In: RoboCup-2000: Robot Soccer World Cup IV, Peter Stone, Tucker Balch and Gerhard Kraetzschmar, editors, LNAI 2019, pp. 29-40, Springer Verlag, Berlim., 2001

[5] L.P Reis and N. Lau, "COACH UNILANG - A Standard Language for Coaching a (Robo)Soccer Team", In: RoboCup-2001: Robot Soccer World Cup V, Andreas Birk, Silvia Coradeshi, Satoshi Tadokoro editors, LNAI 2377, pp. 183-192, Springer Verlag, Berlim 2002

[6] Patrick Riley, "SPADES: System for Parallel Agent Discrete Event Simulation", AI Magazine, 24(2):41–42, 2003

[7] Patrick Riley, "SPADES for Parallel Agent Discrete Event Simulation", http://spades-sim.sourceforge.net/

[8] Smith, Russell. "Open Dynamics Engine v0.039 User Guide", http://opende .sourceforge.net/, 2003

[9] The Expat XML parser, http://expat.sourceforge.net. 2004

[10] Ruby Home Page, http://www.ruby-lang.org, 2004

[11] Boost C++ Libraries, http://www.boost.org, 2004

CLAN - A CAN 2.0B Protocol Controller for Research Purposes

Arnaldo S. R. Oliveira, Nelson L. Arqueiro, Pedro N. Fonseca

Abstract – **The CLAN intellectual property core is a CAN 2.0B controller developed at the Electronics and Telecommunications Department of the University of Aveiro, for research and educational purposes and in particular with the aim of providing the adequate hardware support to implement and validate higher layer protocols such as TTCAN or FTT-CAN. It was modelled at RTL level using the VHDL hardware description language, synthesized, implemented and tested on Xilinx FPGAs. However, the model is technology independent and can be synthesized for different implementation technologies from FPGAs to ASICs. The CLAN IP core fully implements the CAN 2.0B specification and it includes also a synchronous parallel microprocessor interface, interrupt generation logic and some advanced features, such as message filtering, single shot transmission and extended error and statistics logs. The data bus width can be 8, 16 or 32 bits wide. For applications where a microprocessor interface is not needed or a different interface is required, the core internal module that implements the protocol can be used separately. The CLAN controller with microprocessor interface logic occupies about 30% of a Xilinx Spartan-IIE XC2S300E FPGA, corresponding to 100,000 equivalent logic gates, approximately. It was tested with other CAN controllers operating at 1Mbit/seg.**

Resumo – **O módulo CLAN é um controlador CAN 2.0B desenvolvido no Departamento de Electrónica e Telecomunicações da Universidade de Aveiro para fins académicos e em particular com o objectivo de conceber um controlador que proporcione o suporte de hardware adequado à implementação de protocolos de alto-nível, tais como o TTCAN ou o FTT-CAN. O controlador CLAN foi modelado ao nível RTL com a linguagem de descrição de hardware VHDL, implementado e testado em FPGAs da Xilinx. No entanto, é importante referir que o modelo é completamente independente da tecnologia podendo ser sintetizado para diferentes tecnologias, desde FPGAs a ASICs. O controlador CLAN implementa completamente a especificação 2.0B do protocolo CAN e inclui também um interface síncrono paralelo para ligação a um microprocessador, circuito gerador de interrupções, filtros de mensagens e vários contadores erros e registos de estatísticas. O barramento de dados pode ser de 8, 16 ou 32 bits. Para aplicações que não necessitem de um interface com processador ou requeiram outro tipo de interface, o bloco interno que implementa o protocolo pode ser usado separadamente. O controlador CLAN ocupa cerca de 30% de uma FPGA Spartan-IIE XC2S300E da Xilinx, correspondendo a cerca de 100.000 portas lógicas equivalentes e foi testado com outros controladores CAN a operar a 1Mbit/seg.**

Keywords – **CAN, TTCAN, FTT-CAN, Protocol controller**

Palavras chave – **CAN, TTCAN, FTT-CAN, Controlador de protocolo**

I. INTRODUCTION

Defined in the late 80's, CAN (Controller Area Network) [1] found wide-spread acceptance in embedded distributed control systems, from automotive to industrial applications. A CAN overview is out of the scope of this paper. The CAN specification available on the web [1] provides a clear description of the protocol.

In spite of its popularity, the application of CAN in safety-critical systems is, nevertheless, impaired by the event-triggered characteristics of the original definition. In CAN, a node can send a message at any time, provided there is silence on the bus (CSMA): the Medium Access Control mechanisms will handle the resulting collisions. As a consequence, a node sending a message has no guarantee in what concerns the delivery time of that message; depending on the message priority, it may loose contention for several consecutive times, thus postponing the effective sending of the message.

For critical applications, time-triggered systems are preferred, due to their scalability, composability and dependability properties [2]. The last few years saw the outcome of some proposals to improve the time characteristics of CAN (e.g., TTCAN [3], FTT-CAN [4]). These take advantage of the fact that Bosch's and ISO specifications define only layers 2 and (partially) 1 of the ISO OSI model. With major or minor changes on the original definition, these new proposals impose some determinism in the message exchange behavior, namely by allowing a node to send its message at well defined instants in time. This is achieved by properly defining mechanisms in the layers above the original definition.

TTCAN (Time-Triggered Communication on CAN) started in ICC'98, the International CAN Conference, where an expert group, including CiA (CAN in Automation), chip providers, users and academia, joined the ISO TC22/SC3/WG1/TF6. The result was ISO 11898-4, part 4 of the ISO 11898 standard, that specifies time triggered communication on CAN [5].

FTT-CAN (Flexible Time-Triggered Communication on CAN) has been proposed at the University of Aveiro as a mean to merge flexibility and timeliness in CAN systems. The aim is to achieve a communication paradigm that allows systems to be both timely, delivering the messages under the specified time constraints, and flexible, by not requiring the message set to be statically defined during system operation.

II. MOTIVATION AND OBJECTIVES

Both proposals for time triggered operation of CAN are built on top of the existing protocol with little or no mod-

ifications (the aim of FTT-CAN is also to provide timely behavior with standard CAN controllers).

The development of a new communication protocol requires its validation. Although simulation and formal validation play an important role here, they are not, on their own, sufficient. The last step in validation is always field tests, and these have to be performed with hardware devices. These tests should also involve the verification that the adopted solution is better than the alternatives. Ideally, we should have a flexible communication controller that can be programmed to follow some specification and that can be modified. Another issue to test is the robustness of the new protocols, mainly in what concerns fault tolerance. To do this, faults have to be introduced in the system in a controlled and predictable way. Again, we meet the need for a controller that we can modify to our desire.

The above requirements cannot be easily fulfilled with the CAN products commercially available [6] because they are hardwired ASIC products or flexible but expensive synthesizable cores. Thus, a CAN controller was developed based on the *CANSim* simulator [7]. The initial requirements for the CLAN project were the following:

- Complete CAN 2.0B implementation;
- Internal status fully visible to effectively support the implementation of higher layer protocols;
- Enhanced logging capabilities (individual error counters and flags) and statistics logs (message counters);
- Flexible message filtering capabilities;
- Customizable interface - parallel/serial, (a)synchronous, (de)multiplexed buses.

III. ARCHITECTURE

The developed CAN controller fully implements the CAN 2.0B specification. The developed Intelectual Property (IP) block was split in two modules, to separate the logic that implements the protocol from the interface:

- The *CLAN Core* module which implements the CAN 2.0B protocol;
- The *CLAN Controller* module which provides a synchronous parallel interface with non-multiplexed buses.

The interface provided is adequate for integration of the *CLAN Controller* with a processor core into a single FPGA or a System-on-Chip.

A. CLAN Core Module

The *CLAN Core* module contains all the circuits required to implement the Medium Access Control (MAC) and the Logical Link Control (LLC) layers of the CAN 2.0B specification. It can be used separately directly connecting to sensors and/or actuators in a CAN node without a microprocessor. Alternatively, it can also be used as a building block to create a controller with a customized interface.

A.1 Interface Ports

The external interface of the *CLAN Core* module is shown on Figure 1. The ports are divided into the following

Figure 1 - *CLAN Core* module interface.

functional groups: *Synchronization and Initialization* (Table I), *Timing Configuration* (Table II), *Mode Setup* (Table III), *General Status and Statistics* (Table IV), *Transmission and Reception Data* (Table V), *Transmission and Reception Configuration* (Table VI), *Error and Fault Confinement* (Table VII), *Message Filtering Setup* (Table VIII) and *Bus Interface* (Table IX). A short description of each port is given into the tables below.

All interface control and status signals are sampled or switched at the falling edge of the *clk* synchronization clock. The *sampleRxBit* port (Table I) provides a clock signal synchronous with the *Sample Point*. This signal combined with the frame state available on the *busFrameState* port is useful for clock synchronization on TTCAN implementations. The single shot transmission capability provided is useful for disabling automatic message retransmission in case of an error within TTCAN and FTT-CAN synchronous windows. The values applied to the *Timing Configuration* ports (Table II) must be stable to ensure a correct operation of the core.

Name	Type	Description
reset	In	Asynchronous reset input
clk	In	Main synchronization signal
clkEnable	In	Enable input for the core "clk" signal
sampleRxBit	Out	Clock signal synchronous with the Sample Point

Table I
SYNCHRONIZATION AND INITIALIZATION PORTS.

Name	Type	Description
timeSeg1	In	"Length - 1" of Time Segment 1 (in time quanta)
timeSeg2	In	"Length - 1" of Time Segment 2 (in time quanta)
syncJumpWidth	In	Synchronization Jump Width value (in time quanta)
baudRPScaler	In	Baud Rate Pre-Scaler value used for "clk" frequency division

Table II
TIMING CONFIGURATION PORTS.

Name	Type	Description
singleShotTx	In	When active disables automatic message retransmission in case of error
multiSample	In	Sampling mode for improved noise imunity
monitorMode	In	When active sets the output driver permanently to "recessive" level

Table III
MODE SETUP PORTS.

Name	Type	Description
rxtxMode	Out	Current RX/TX mode (None, Rx, Tx, Arbitration)
busFrameState	Out	Current state of the frame present on the bus
arbitrationLost	Out	Activated during one CAN bit time in case of arbitration lost
arbitLostBitIdx	Out	When "arbitrationLost" = 1 this output indicates the bit where arbitration was lost
txMsgCount	Out	Number of successfully transmitted messages
rxMsgCount	Out	Number of successfully received messages
clrMsgCounters	In	When activated clears the "txMsgCount" and "rxMsgCount" counters
busActivityFlag	Out	Indicates the presence of bus activity

Table IV
GENERAL STATUS AND STATISTICS PORTS.

Name	Type	Description
txMsgIdExt	In	Tx message extended identifier flag
txMsgId	In	Tx message identifier
txMsgRTR	In	Tx message RTR flag
txMsgDLC	In	Tx message DLC value
txMsgData	In	Tx message data bytes
rxMsgIdExt	Out	Rx message extended identifier flag
rxMsgId	Out	Rx message identifier
rxMsgRTR	Out	Rx message RTR flag
rxMsgDLC	Out	Rx message DLC value
rxMsgData	Out	Rx message data bytes

Table V
TRANSMISSION AND RECEPTION DATA PORTS.

A.2 Internal Structure

The internal structure of the *CLAN Core* module is shown on Figure 2. A short description of each block is given into the following subsections.

Bit Stuffing Unit

The *Bit Stuffing Unit* is used to:

- insert stuff bits on the transmitted bit stream;
- check and remove stuff bits from the received bit stream.

The *Bit Stuffing Unit* is shared by the transmission and reception parts of the core, because unless an error has occurred or the transmitter looses arbitration, within the stuffed fields the transmitted and the received bits should match.

CRC Unit

The *CRC Unit* calculates and checks the CRC sequence included in the frame. Similarly to the *Bit Stuffing Unit*, it is shared among the transmission and reception parts of the controller. In transmit mode, it calculates the CRC sequence during the *Start of Frame*, *Arbitration*, *Control* and *Data fields*. During the *CRC Sequence* field, the calculated sequence is shifted into the bus. In reception mode it compares the received sequence with the locally computed sequence in order to detect errors on the received bit stream.

Reception Unit

The *Reception Unit* latches the bus bit at the *Sample Point* and performs the de-serialization of the reception bitstream. It also acknowledges a correctly received frame during the *Acknowledge Slot* field.

Transmission Unit

The *Transmission Unit* performs the serialization of the message to send, determining the bit to be transmitted by

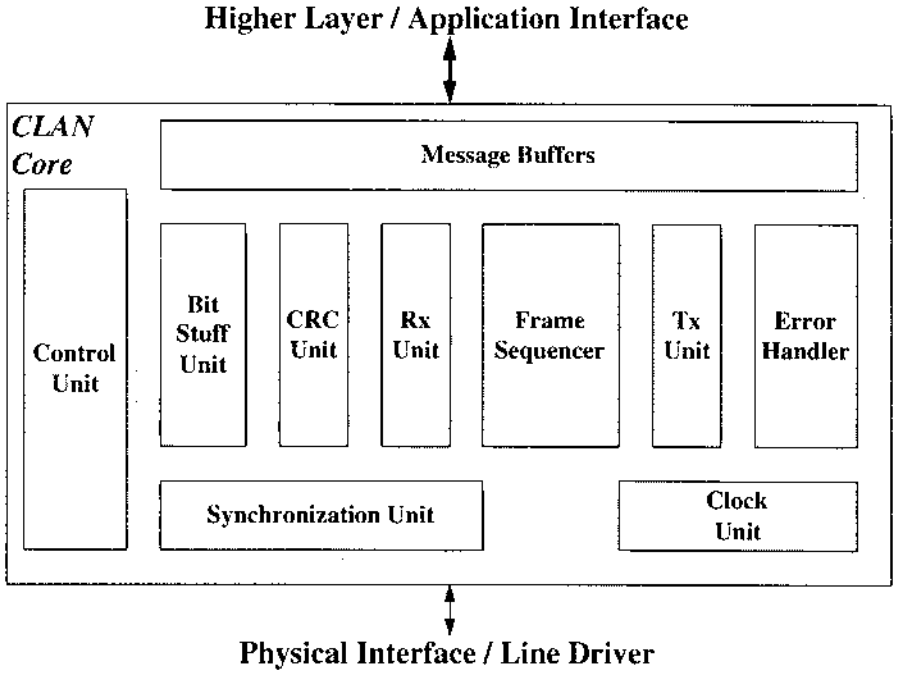


Figure 2 - *CLAN Core* internal block diagram.

the node and setting it at the beginning of the bit time. The sources for the transmitted bit are the following:

- A message bit from the *ID*, *RTR*, *DLC* or *DATA* fields;
- A stuff bit;
- A *CRC* bit;
- A fixed polarity bit (*recessive* or *dominant*);
- An acknowledge bit generated by the *Reception Unit*;
- An error frame bit produced by the *Error Handler*.

Frame Sequencer

The *Frame Sequencer* plays a central role within the controller, performing the following tasks:

- Arbitration;
- Accepting requests to transmit messages;
- Detecting a *Start of Frame* field on the bus;
- Sequencing fields in *Data*, *Overload* and *Remote Transmission Request* frames;
- Signaling the successful transmission of a message and the end of a message reception;
- Responding to *Overload* frames.

Figure 3 shows a partial behavioral specification of the *Frame Sequencer*. It consists of two parallel state machines: the *Rx/Tx Mode State Machine* and the *Frame State Machine*. The former defines the operating mode of the controller. The last establishes the sequence of fields for all frame types except *Error* frames, which are generated directly by the *Error Handler* and applied to the *Transmission Unit*.

Error Handler

The *Error Handler* performs all activities related to fault confinement, error detection, counting and signaling. Internally it implements the mechanisms to detect the different error types and the error counters specified on the standard.

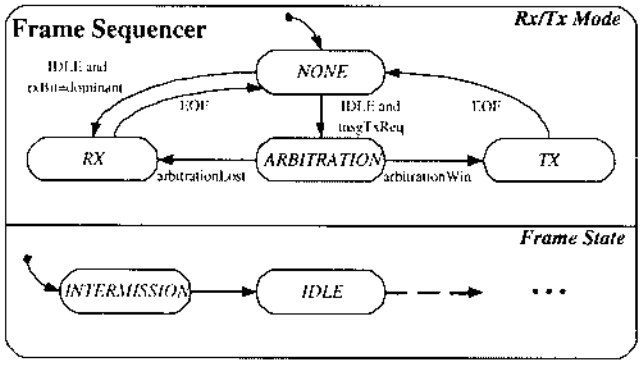


Figure 3 - Partial behavioral specification of the *Frame Sequencer* module.

When an *Error* frame has to be sent, the transmission is performed through the *Transmission Unit* and the *Frame Sequencer* is disabled until the end of the *Error* frame.

Message Buffers

As the name implies, the *Message Buffers* are used to store messages. Two buffers are accessible from the outside of the module: one for transmission and the other for reception. However, internally the *Transmission* and *Reception* units contain shift registers for message serialization and de-serialization that act as temporary buffers.

Clock Unit

The *Clock Unit* generates all the clocks required to control and synchronize the activities of the other core components. Behavioral modelling of the CAN controller has shown that two clock signals are required for such purposes [7]: a *Synchronization Clock* with frequency f_{SYNC} and *Control Clock* with frequency f_{CTRL}:

$$f_{SYNC} = \frac{1}{T_Q}$$

Name	Type	Description
reqMsgTx	In	When activated, requests the transmission of the message applied to the txMsg(IdExt, Id, RTR, DLC, Data) ports
cancelTxReq	In	When activated, cancels the previous transmission request, if still pending
msgTxStarted	Out	Activated during one CAN bit time at the start of a message transmission
msgTxOk	Out	Activated during one CAN bit time at the end of a successful message transmission
msgTxFailed	Out	Activated during one CAN bit time when a message transmission fails
rxMsgReady	Out	Activated during one CAN bit time at the end of a successful message reception
releaseRxBuffer	In	When activated, releases the Rx buffer, allowing the core to write a new received message on the buffer accessible trough the rxMsg(IdExt, Id, RTR, DLC, Data) ports
dataOverrunError	Out	Activated when a new message was received before an external release of the Rx buffer containing the previous received message. The newly message received is discarded

Table VI
TRANSMISSION AND RECEPTION CONFIGURATION PORTS.

$$f_{CTRL} = 2 \cdot f_{SYNC}$$
$$f_{CLK} = (baudRPScaler + 1) \cdot f_{CTRL}$$

where T_Q is the *Time Quantum* period and f_{CLK} is the *clk* frequency. It means that for a given *Time Quantum* value, an input clock with only twice the frequency is needed.

Control Unit

The *Control Unit* generates all signals that control the other units, mainly enable and reset signals. Figure 4 shows a simplified view of the core internal control sequence within a CAN bit time. The frequency divider that generates the *Control Clock* signal is triggered by the falling edge of the *clk* input. The majority of the units are triggered by the rising edge of the *Control Clock* and during the *Time Segment 2*, i.e. after the *Sample Point*. This imposes some restrictions on the duration of the *Time Segment 2*, namely its minimum duration must be 2 *Time Quanta*. This constraint is required to decrease the number of internally generated clock signals and to limit the frequency of the clock

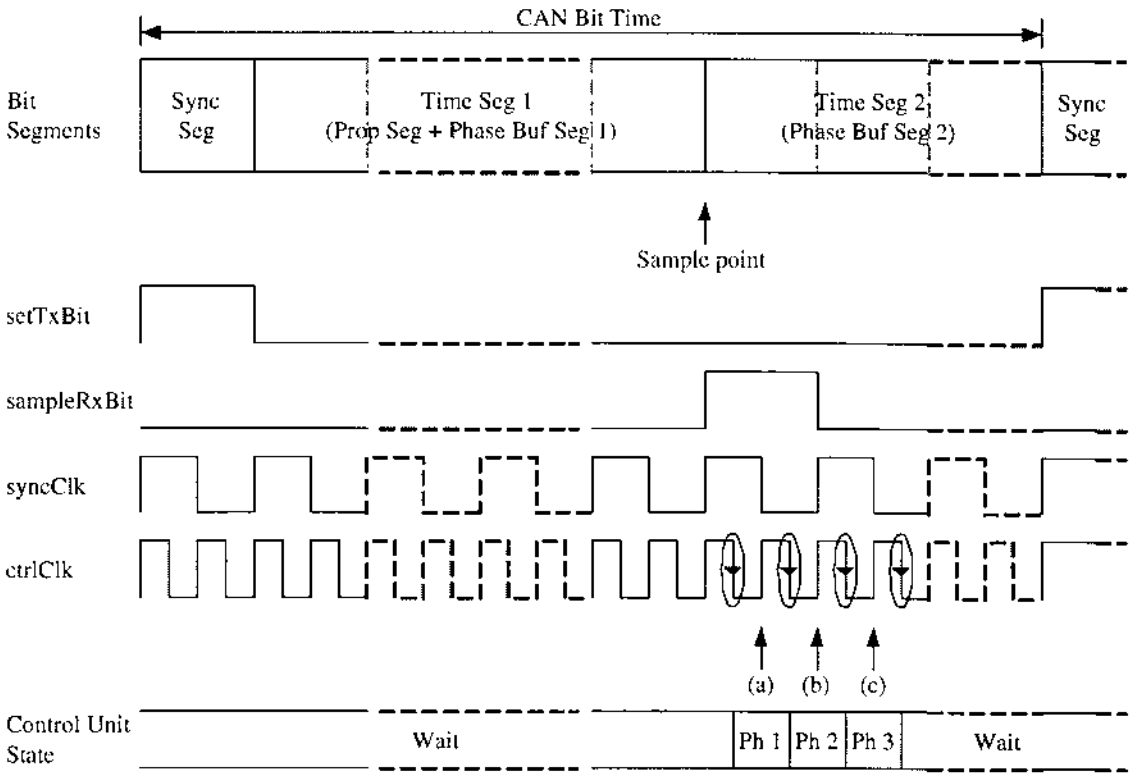
Name	Type	Description
configError	Out	Activated when the timing parameters are invalid
faultConfinState	Out	Current fault confinement state ("Error Active", "Error Passive", "Bus Off")
faultConfStEvent	Out	Activated during one CAN bit time after a change on the fault confinement state
bitErrorFlag	Out	Active during one CAN bit time in case of a bit error
stuffErrorFlag	Out	Active during one CAN bit time in case of a stuff error
crcErrorFlag	Out	Active during one CAN bit time in case of a CRC error
formErrorFlag	Out	Active during one CAN bit time in case of a form error
ackErrorFlag	Out	Active during one CAN bit time in case of a acknowledge error
errorDetected	Out	Active during one CAN bit time in case of a bit, stuff, CRC, form or acknowledge error
txErrorCount	Out	Tx Error Count as defined on the CAN specification
rxErrorCount	Out	Rx Error Count as defined on the CAN specification
errorWarnLimit	In	Threshold value used to flag a disturbed bus
errorWarnFlag	Out	Activated when one of the error counters is greater than the "errorWarnLimit" value
bitErrorCount	Out	Number of bit errors occurred
stuffErrorCount	Out	Number of stuff errors occurred
crcErrorCount	Out	Number of CRC errors occurred
formErrorCount	Out	Number of form errors occurred
ackErrorCount	Out	Number of acknowledge errors occurred

Table VII
ERROR AND FAULT CONFINEMENT PORTS.

applied to the core for a given transmission rate. However, it is important to note that this restriction is compliant with the maximum duration of the *Information Processing Time* defined on CAN specification.

Synchronization Unit

The *Synchronization Unit* generates the *sampleRxBit* and the *setTxBit* clocks used to latch the reception and transmission signals at the correct time instants, based on bus transitions and on the timing parameters of the node. The period of the CAN bit is given by the following expression:



Edges responsible for control unit state transitions

(a) - Frame sequencer update phase; Error handler detection phase

(b) - Rx buffer write; CRC unit enable; Bit stuff unit enable; Frame sequencer execution phase; Error handler update phase

(c) - Tx buffer read; Error handler reaction phase

Figure 4 - *CLAN Core* internal control sequence.

Name	Type	Description
rangeFiltering	In	When activated, filtering is performed based on the lower and upper bound of message identifiers specified by the next two ports; when deactivated, filtering is based on identifier patterns
accMskLoBound	In	Specifies the identifier lower bound or don't care bits of the identifier used for message filtering
accIdUpBound	In	Specifies the identifier upper bound or significant bit values of the identifier used for message filtering

Table VIII
MESSAGE FILTERING SETUP PORTS.

Name	Type	Description
busBit	In	Current bus level detected by the input transceiver
rxBit	Out	Bus level at the previous sample point
txBit	Out	Current level applied to the output transceiver

Table IX
BUS INTERFACE PORTS.

where T_{CLK} is the period of the external clock applied to the core. The period T_{CTRL} of the control clock is:

$$T_{CTRL} = T_{CLK} \cdot (baudRPScaler + 1)$$

The values of the timing parameters must respect the following relation:

$$timeSeg1 > timeSeg2 > syncJumpWidth$$

$$T_{BIT} = T_{CLK} \cdot 2 \cdot (baudRPScaler + 1) \cdot (timeSeg1 + timeSeg2 + 3)$$

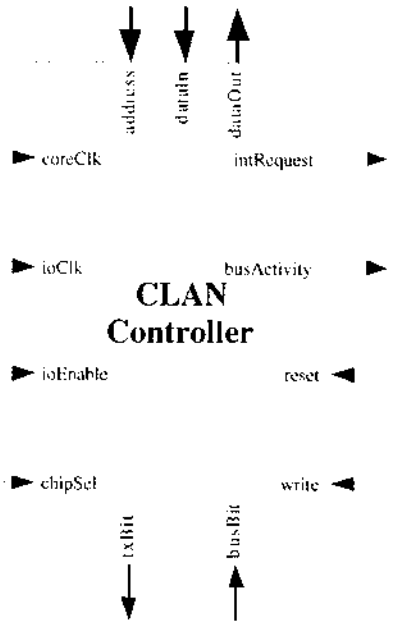


Figure 5 - *CLAN Controller* module interface.

otherwise the *configError* output will be active and the core will remain in the reset state.

B. CLAN Microprocessor Interface Module

Based on the *CLAN Core* module, different interfaces can be created. The first interface built was a synchronous parallel interface with a data bus of 8, 16 or 32 bits.

B.1 Interface Ports

The external interface of the *CLAN Controller* module is shown on Figure 5. A short description of each port is given on Table X. The *coreClk* is the *clk* synchronization signal of the *CLAN Core* module (Table I). Figure 6 shows examples of read and write cycles. The microprocessor changes the signals at the falling edge of the *ioClk* signal. The *CLAN Controller* validates the signals at the rising edge of the same clock. Thus, if the *ioClk* frequency is adequate for internal core synchronization, the *coreClk* and *ioClk* clock signals can be connected together to the same clock source.

B.2 Configuration Registers

The configuration registers map into a space of 128 addresses all the input and output ports of the *CLAN Core* module. All registers are at a fixed offset location independently on the bus width (Table XI). The complete description of the configuration registers can be found at the *CLAN Project Web Page* [8].

IV. MODELLING AND SIMULATION

The CLAN IP block was modelled with the VHDL hardware description language because VHDL provides the adequate abstractions to model the CAN controller building blocks, such as multiplexors, registers, state machines, etc. The model created contains about 3700 lines of code and it is completely independent of the implementation technology. It was successfully validated with the ModelSim VHDL simulator.

Name	Type	Description
reset	In	Asynchronous reset input
coreClk	In	Core internal synchronization signal
ioClk	In	Interface synchronization signal
chipSel	In	Global interface enable signal
ioEnable	In	Enable signals for individual bytes of a multi byte data bus interface
write	In	Write enable signal
address	In	Address bus
dataIn	In	Data input bus
dataOut	Out	Data output bus
intRequest	Out	Interrupt request output for microcontroller
busActivity	Out	Flag that indicates activity on the bus
busBit	In	Port for connection to the reception transceiver (line-driver)
txBit	Out	Port for connection to the transmission transceiver (line-driver)

Table X
CLAN Controller PORTS.

V. SYNTHESIS, IMPLEMENTATION AND TEST

The CLAN IP block was synthesized and implemented on a Xilinx XC2S300 Spartan-IIE low cost FPGA. The synthesis report is shown on Figure 8. The complete circuit occupies about 30% of the available slices (logic cells) corresponding to 100,000 logic gates. The core internal logic can operate up to 42MHz. Figure 7 shows the complete project hierarchy. To use the CLAN IP block as a black box in a project three components must be included:

- The file containing the synthesized netlist;
- The file *CAN.VHD* containing a package with generic CAN definitions;
- The file *CLANPublic.vhd* containing a package with CLAN specific definitions.

The *CLAN Core* was tested within a bus with other commercial CAN controllers operating at 1Mbit/seg. The test setup is depicted on Figure 9. The main purpose of this setup is to perform a simple functional validation of the controller that must retransmit all received messages.

The *CLAN Controller* module was also integrated on the ARPA System-on-Chip with a MIPS32 processor optimized for real-time systems [9]. An API library was developed that allows the configuration and communication with the *CLAN Controller* from a program in C language.

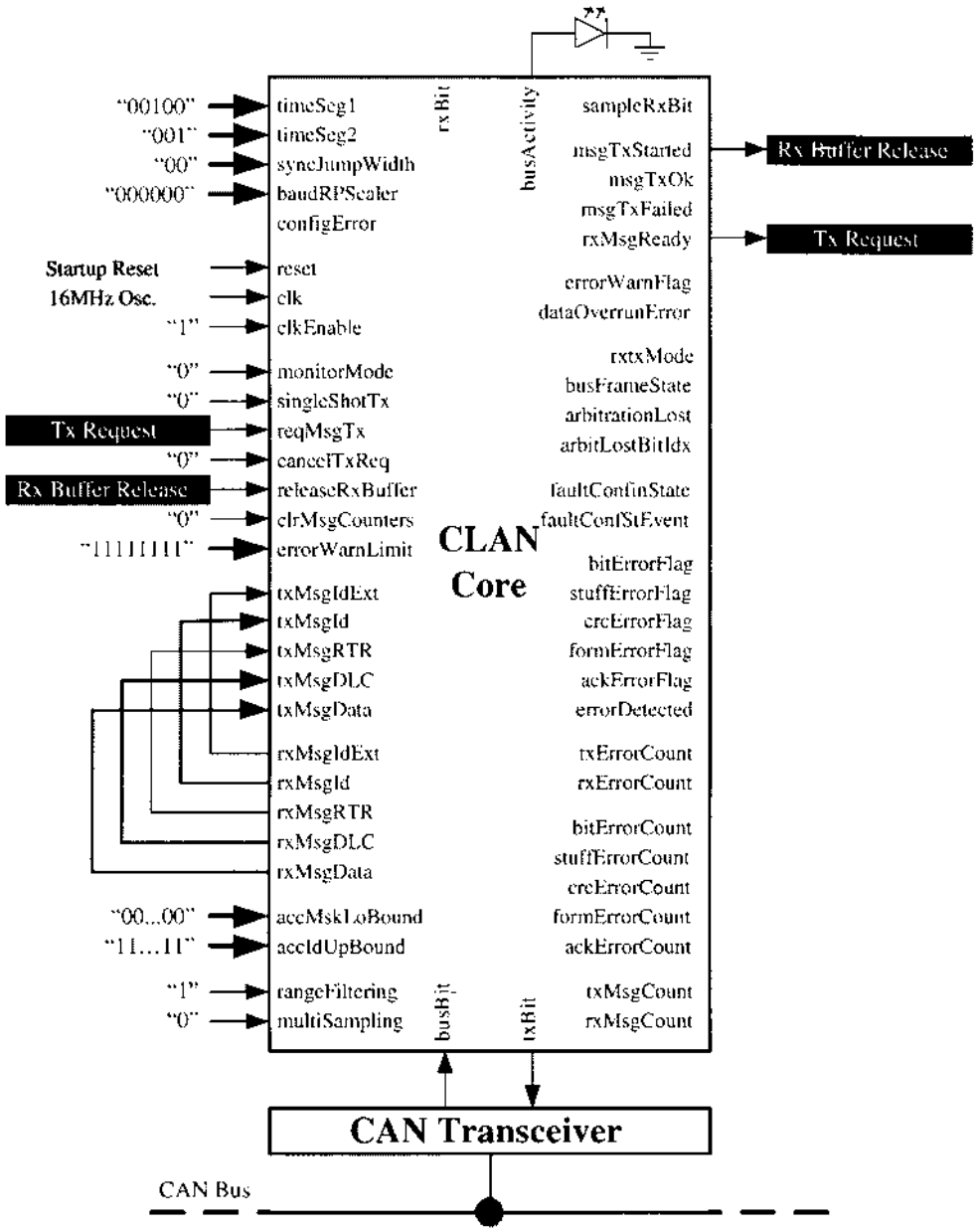


Figure 9 - *CLAN Core* loop-back test setup.

VI. CONCLUSION

A full CAN 2.0B controller with synchronous parallel microprocessor interface was presented in this paper. The IP core was developed for educational and research purposes. It communicates correctly with other commercial controllers operating up to 1Mbit/seg. However, it is important to refer that it was not validated with the CAN conformance tests. The web page of the CLAN project with detailed and updated information can be found at [8].

REFERENCES

[1] Robert Bosch GmbH, "CAN Specification Version 2.0 (http://www.can.bosch.com)", 1991.

[2] Hermann Kopetz, *Real-Time Systems: Design Principles for Distributed Embedded Applications*. Kluwer International Series in Engineering and Computer Science. Kluwer Academic Publishers, 1. edition, 1997.

[3] Thomas Fuhrer, Bernd Muller, Werner Dieterle, Florian Hartwich, Robert Hugel, and Michael Walther, "Time triggered communication on CAN (Time Triggered CAN - TTCAN)", in *ICC 2000 - 7th International CAN Conference*. October 2000, CiA - CAN in Automation.

[4] Luís Almeida, Paulo Pedreiras, and José Alberto Fonseca, "The FTT-CAN protocol: Why and how", *IEEE Transactions on Industrial Electronics*, vol. 49, no. 6, pp. 1189–1201, December 2002.

[5] ISO/TC 22/SC 3/WG 1, "Road Vehicles — Controller Area Network (CAN) Part 4: Time Triggered Communication", Tech. Rep. ISO/WD 11898-4, ISO, December 2000.

[6] CAN in Automation, "CAN products web page (http://www.can-cia.org)", 2005.

[7] Arnaldo Oliveira, Pedro Fonseca, Valery Sklyarov, and António Ferrari, "An object-oriented framework for can protocol modeling and simulation", in *FET 2003 - The 5th IFAC International Conference on Fieldbus Systems and their Applications*, July 2003, pp. 243–248.

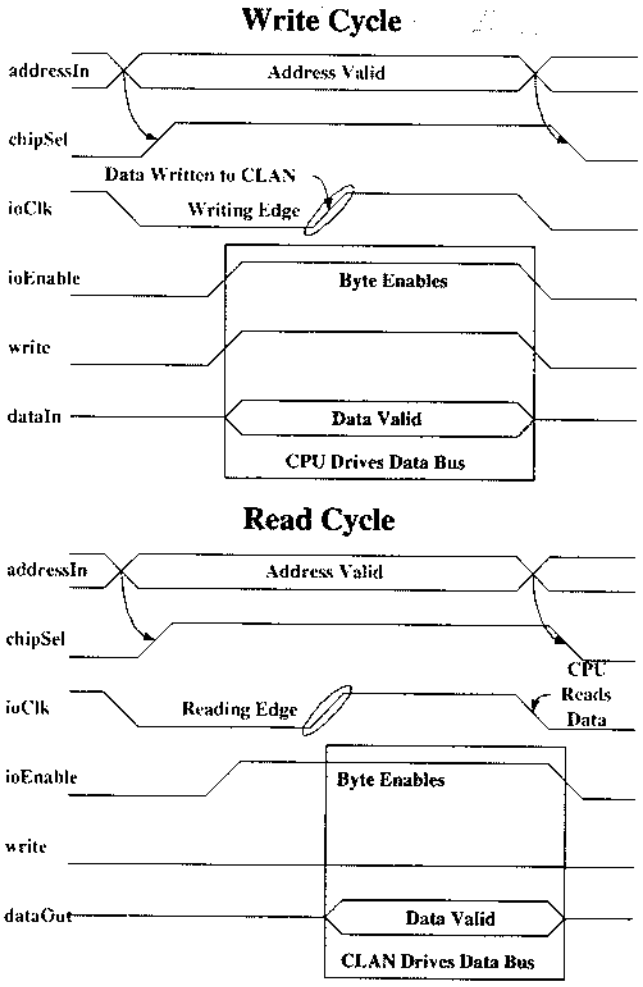


Figure 6 - *CLAN Controller* write and read bus cycles.

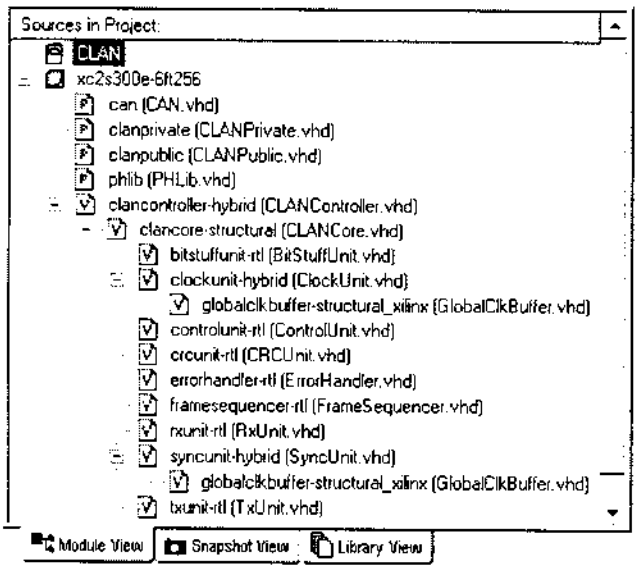


Figure 7 - CLAN Project hierarchy.

Offset (Hex)	Register Name	Access Type
00h	Command	W
00h	Status 0	R
04h	Status 1	R
08h	Control	RW
0Ch	Rx/Tx Status	R
10h	Arbitration Lost Capture	R
14h	Error Status	R
18h	Bus Timing	RW
1Ch	Interrupt Enable	RW
20h	Interrupt Identification	R
24h	Rx Error Count	R
28h	Tx Error Count	R
2Ch	Error Warning Limit	RW
30h	Bit Error Count	R
34h	Stuff Error Count	R
38h	CRC Error Count	R
3Ch	Form Error Count	R
40h	Acknowledge Error Count	R
50h	Acceptance Mask/Lower Bound	RW
54h	Acceptance Identifier/Upper Bound	RW
58h	Rx Message Count	R
5Ch	Tx Message Count	R
60h	Rx Message Control	R
64h	Rx Message Identifier	R
68h	Rx Message Data 03	R
6Ch	Rx Message Data 47	R
70h	Tx Message Control	RW
74h	Tx Message Identifier	RW
78h	Tx Message Data 03	RW
7Ch	Tx Message Data 47	RW

Table XI

CLAN Controller REGISTER NAMES AND OFFSETS.

```
Final Synthesis Report
=========================================================
Device utilization summary:
Selected Device : 2s300eft256-6
  Nº of Slices:              931 out of 3072  ( 30%)
  Nº of Slice Flip-Flops:    863 out of 6144  ( 14%)
  Nº of 4 input LUTs:       1513 out of 6144  ( 24%)
  Nº of TBUFs:                32 out of 3072  (  1%)
  Nº of GCLKs:                 2 out of    4  ( 50%)
Timing Summary:
  Speed Grade: -6
  Min. period: 23.3ns (Max. Frequency: 42.8MHz)
  Min. input arrival time before clock:  11.4ns
  Max. output required time after clock: 10.2ns
  Maximum combinational path delay:       3.8ns
```

Figure 8 - Summary of *CLAN Controller* synthesis report.

[8] Arnaldo S. R. Oliveira, "CLAN project web page (http://www.ieeta.pt/~ arnaldo/projects/clan/)", 2005.

[9] Arnaldo S. R. Oliveira, Valery A. Sklyarov, and António Ferrari, "The ARPA project - creating an open source real-time system-on-chip",

Electrónica e Telecomunicações, vol. 4, no. 3, pp. 389-392, September 2004.

Implementação do jogo Minesweeper usando a linguagem Handel-C

Leonel Neves

Resumo – **Pretendeu-se criar uma versão do famoso jogo do Microsoft Windows® "Minesweeper" para a placa RC100 da Celoxica, suportado por um ambiente gráfico cromaticamente rico (16bpp), organizado em hierarquia de janelas e *event driven*, à semelhança dos actuais sistemas operativos.**

O programa começa por copiar os bitmaps da memória Flash para o bloco de SSRAM que é usado para dados, seguindo-se a inicialização das estruturas de dados do jogo. A partir desse momento, vão correr em paralelo os processos: *video driver*, *mouse driver*, escrita no ecrã e leitura do rato, gerador de números aleatórios, contadores de tempo, *window manager* e o jogo propriamente dito.

Optou-se por usar um dos blocos de SSRAM para conter os *pixels* que são continuamente alimentados à saída VGA. São lidos dois *pixels* de cada vez num ciclo de relógio, o que liberta o ciclo de relógio seguinte para que se possa escrever nova informação nessa RAM, se necessário.

O ponteiro do rato é definido numa ROM constituída por um *array* de valores indexados a um *colormap* de cores a 24bpp. Por motivos estéticos, resolveu fornecer-se sombra ao ponteiro, a qual está definida numa ROM adicional.

Poder-se-á, sem grande dificuldade de adaptação, utilizar este ambiente para correr qualquer programa para o referido *hardware*.

Abstract – **This article describes an implementation of the well-known Microsoft Windows® game "Minesweeper", on the basis of the Celoxica RC100 board. The game runs in a rich and colorful environment (16bpp), which is event driven and organized as a window hierarchy, just like modern operating systems.**

At the beginning, the program copies bitmaps previously stored in the Flash RAM to the SSRAM block used as a databank. Then, the game structures are initialized. After that, several processes run in parallel: the video driver, the mouse driver, output to screen, mouse processing, random numbers generator, time counters, window manager and the game itself.

One of the SSRAM blocks is used exclusively as video RAM, thus providing a continuous feed of pixels to the VGA output. In a single clock cycle two pixels are fetched, which frees the following clock cycle, making it possible to write new data into that RAM block.

The mouse pointer is defined as an array of indexes to a 24bpp colormap and is enhanced by a mild and pleasant shadow, also defined as an array. Both pointer and shadow are kept in ROMs.

This environment can be tailored to run any program for the RC100 hardware without to much effort.

1. INTRODUÇÃO

Os utilizadores de sistemas digitais são cada vez mais exigentes relativamente à qualidade das interfaces Humano-Computador. Hoje em dia já é possível apreciar o elevado nível de complexidade do grafismo dos mais variados sistemas electrónicos, da indústria dos telefones celulares às consolas de jogos. Neste contexto, faz sentido pensar em soluções de visualização igualmente agradáveis destinadas ao uso de sistemas reconfiguráveis autónomos.

No decorrer do desenvolvimento deste trabalho, embora o produto final seja um simples jogo, pretendeu-se dar ênfase à capacidade de simular um ambiente gráfico de alto nível numa pequena placa electrónica. Para tal, foram exploradas as características específicas do *hardware*, como a capacidade de suportar processamento paralelo real, uma das vantagens dos sistemas que utilizam uma FPGA relativamente aos que usam um PIC (Processor Integrated Circuit, microcontrolador). Assim, de entre as linguagens disponíveis para especificar sistemas digitais, optou-se pelo Handel-C por ser de alto nível. O programa destina-se a correr na placa RC100 da Celoxica e foi desenvolvido com o *software* DK2 da Celoxica. No site [1] encontra-se informação detalhada sobre a linguagem, o *software* e a placa utilizados.

A. Objectivos

O trabalho desenvolvido pretendeu, entre outros, responder aos seguintes desafios:

- disponibilizar um ambiente gráfico de alto nível para o jogo Minesweeper;
- construir uma plataforma adequada à solução pretendida e ao *hardware*;
- responder com sucesso à especificidade dos dispositivos periféricos rato e monitor VGA;
- integrar técnicas de optimização e de reutilização dos recursos da FPGA;
- utilizar eficientemente os blocos de SSRAM e a Flash RAM.

B. Descrição global do trabalho

A versão do jogo Minesweeper aqui implementada consiste numa matriz de 16x16 células clicáveis, como se pode ver na figura 1. No início de um novo jogo, são colocadas automaticamente 40 minas ocultas na área virtual correspondente às células, recorrendo a um gerador

de números aleatórios. Para dar início ao jogo, clica-se numa célula qualquer, momento em que o tempo começa a contar. Garante-se ao jogador que a primeira célula descoberta nunca tem mina. Cada célula descoberta contém a indicação do número de minas nas células vizinhas ou, se contiver uma mina, o jogo termina e o jogador perde. Se a célula clicada estiver numa área sem minas, o processo de descoberta é automático, de forma recursiva, para toda essa área. O jogador assinala uma mina clicando com o botão direito do rato na célula suspeita. O jogador deve descobrir todas as minas antes de se atingir o tempo limite, 999 segundos. O jogo termina com sucesso quando todas as minas, e apenas essas, estão assinaladas correctamente. Um novo jogo pode ser iniciado a qualquer momento clicando no botão que contém o desenho de um smiley. A intervenção do utilizador é feita através de um rato directamente ligado à placa. A figura 2 ilustra a interligação das componentes de *hardware* utilizadas.

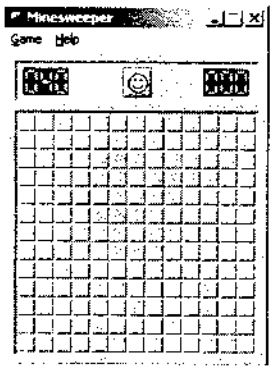


Fig. 1 – Minesweeper

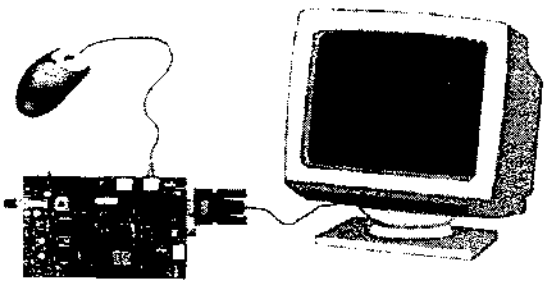


Fig. 2 – Interligação da placa RC100 e periféricos

O único dispositivo de interacção disponível ao utilizador é o rato, pois é esse que é usado quando se joga num computador.

II. ORGANIZAÇÃO DO PROGRAMA

Este programa faz uso intensivo de bitmaps, os quais já devem estar na memória Flash da placa (a partir do endereço 0) antes de sua execução. No início, o programa copia os bitmaps da Flash para o bloco 1 da SSRAM. De seguida, inicializa as estruturas de dados do jogo. É então que arrancam em paralelo os processos:

- video driver;
- mouse driver;
- escrita no ecrã e leitura do rato;
- gerador de números aleatórios;
- contadores de tempo;
- window manager e o jogo propriamente dito.

Todos estes processos são, basicamente, ciclos infinitos independentes. A comunicação entre eles, como se verá mais adiante, efectua-se ou através da RAM ou através de variáveis globais. Esta é a parte activa da função *main*:

```
copyFlash2RAM();
setupNewGame();
par { // execução em paralelo:
  RC100VideoDriver(&Video);
  RC100PS2MouseDriver(&Mouse, RC100_MOUSE_PORT);
  runScreenAndMouse();
  runRandomGen();
  runTimer();
  runWindowManagerAndGame();
}
```

III. MODELAÇÃO E ESTRATÉGIAS ADOPTADAS

A. Suporte para a interface gráfica

Para se conseguir libertar o jogo da necessidade de estar sincronizado com o feixe de varrimento do ecrã, optou-se, logo no início do projecto, por utilizar o bloco 0 de SSRAM exclusivamente como RAM de vídeo. Para obter gráficos de elevada qualidade e ainda manter disponível uma elevada largura de banda para a escrita de pixels na RAM, para além da obtida nos períodos de *blanking* do feixe de varrimento, resolveu guardar-se em cada endereço, de 32 bits, a informação de 2 pixels, cada um com 16 bits distribuídos numa variante da forma RGB565. Na macro `runScreenAndMouse()`, sempre que o feixe de varrimento está nas colunas ímpares, é lido um endereço, tendo o cuidado de verificar situações excepcionais como o fim visível das linhas e o fim visível do ecrã. Os pixels correspondentes são armazenados num *buffer* (a variável `video_data`), o qual é consultado no momento de enviar um pixel para a saída VGA após ser convertido para RGB888. Por sua vez, quando invocada no decorrer do jogo, a função `writePixel(x, y, cor)` faz a escrita na RAM tendo em conta o processo de leitura. Assim, o pixel de coordenadas (X,Y) é mapeado no endereço N = X / 2 + Y x BG_WIDTH / 2, em que BG_WIDTH é o número de pixels por linha, ficando o pixel guardado na parte baixa ou alta em função do bit 0 de X, aquele que indica se a coluna é par ou ímpar (figura 3).

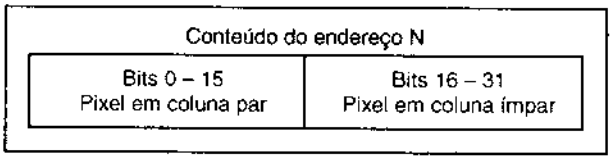


Fig. 3 – Conteúdo de um endereço da RAM de vídeo

Como não pode acontecer mais do que um acesso à RAM em cada ciclo de relógio, a macro `runScreenAndMouse()` controla a variável global `vram_ocupada`, a qual é consultada em `writePixel(...)` antes desta escrever o pixel. De notar que a escrita de um pixel consiste em ler o conteúdo de um endereço de memória, combinar os bits

lidos com os bits novos (simples substituição) e, por fim, escrever o resultado no mesmo endereço de memória.

O ponteiro do rato não é armazenado na RAM de vídeo. A sua apresentação no ecrã é conseguida fazendo a leitura de um pixel do *buffer* e combinando-o com um valor lido da matriz de pontos do cursor, definida estaticamente. De seguida afecta-se cada componente de cor do pixel com um coeficiente de transparência, cujo valor vem de uma matriz estática, o qual fornece ao ponteiro uma sombra visualmente agradável, 75% transparente. As coordenadas do ponteiro do rato e o estado dos botões são armazenados numa estrutura de dados (a variável global `mouse_data`) adequada às necessidades do programa. De notar que não é obrigatório o *hot spot* do cursor coincidir com o canto superior esquerdo da respectiva matriz, o que oferece diversas possibilidades criativas. A matriz contém índices para uma *look-up table* com as cores que o cursor pode apresentar, já em RGB888. Ambas as matrizes referidas são constituídas por simples vectores de valores.

B. Interface gráfica orientada aos eventos

As interfaces gráficas *event driven* pressupõem a existência de uma estrutura computacional capaz de analisar o estado do sistema a intervalos mais ou menos regulares e identificar as diferenças entre dois estados consecutivos. Estas diferenças são os eventos. A figura 4 ilustra este processo, o qual é implementado no procedimento macro `runWindowManagerAndGame()`.

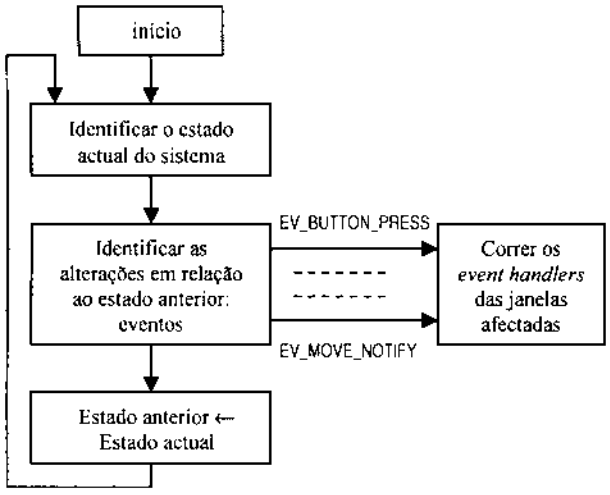


Fig. 4 – Mecanismo de base da gestão por eventos

Os eventos podem ser, por exemplo, movimentos do rato, botões pressionados e entrada do cursor numa janela. Por sua vez, uma janela é uma entidade conceptual que corresponde a uma área rectangular do ecrã e têm associada um conjunto de atributos, incluindo uma lista de sensibilidade a eventos. As janelas são organizadas numa hierarquia, a começar na janela de nível inferior, chamada de raiz, que cobre todo o ecrã e da qual todas as outras descendem. Ao nível da implementação, esta organização tem por base a estrutura `Window` que tem todos os campos necessários à execução do programa:

```
typedef struct {
  unsigned 3 id;        /* id da janela          */
  unsigned 3 idparent; /* id da janela pai      */
  unsigned 12 wx, wy;  /* posição rel. ao pai */
  unsigned 12 wx_abs, wy_abs;  /* pos. abs.    */
  unsigned 12 wwidth, wheight; /* dimensões    */
  void (*action[NUM_EVENTOS])(Event*);
                    /* lista de sensibilidade */
} Window;
```

A parte do programa que faz a gestão das janelas chama-se, apropriadamente, gestor de janelas (*window manager*). As estruturas `Event`, `Window` e `WManager` contêm a informação relevante neste contexto. Estes conceitos estão bastante desenvolvidos em "Xlib Programming Manual" [2], dos quais este trabalho apenas implementa um subconjunto muito pequeno. Normalmente os mecanismos de geração de eventos e de gestão de janelas são independentes, mas dada a necessidade de optimizar os recursos da FPGA, optou-se por juntá-los.

C. O jogo Minesweeper

Uma vez definida a plataforma computacional sobre a qual corre o jogo e por observação do jogo original (figura 1) foram identificadas as áreas funcionais mapeáveis em janelas. A figura 5 contém a localização e a identificação das janelas:

1. janela do jogo;
2. contador de minas;
3. contador de tempo;
4. botão para reiniciar;
5. matriz de 16x16 células com minas escondidas.

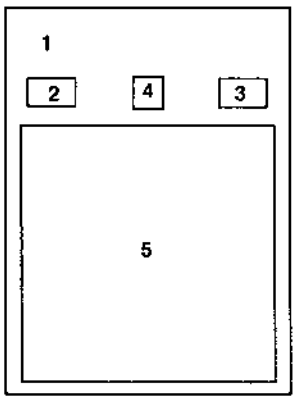


Fig. 5 – Áreas funcionais

Estas áreas correspondem à hierarquia que começa na janela raiz (índice 0 por definição), sobre a qual existe a janela do jogo (índice 1) e, sobre esta, as restantes. Os *event handlers* foram escritos de acordo com as funções a desempenhar no decorrer do jogo, de acordo com as acções do utilizador. Por exemplo, para o botão de início de jogo (janela índice 4) foram implementados:

```
static void smileyPressEV(Event *ev);
static void smileyReleaseEV(Event *ev);
static void smileyEnterEV(Event *ev);
static void smileyLeaveEV(Event *ev);
```

Por defeito, é invocada a função `nullFunc` quando uma determinada acção não está incluída na lista de sensibilidade da janela. Foram também escritos os procedimentos macro e as funções necessárias à implementação das regras do jogo descritas anteriormente. Por exemplo, quando é necessário escrever o valor de uma

célula na respectiva matriz, guardada na RAM de dados, invoca-se a seguinte função:

```
static void setCellValue(unsigned 4 i,
                         unsigned 4 j,
                         unsigned 4 val)
{
   RC100WriteSSRAM1(CELL_DATA_ADDR+adju(i,j,
                    AddressWidth),
                    adju(val, DataWidth));
}
```

Como se pode ver, teve-se o cuidado de não armazenar na FPGA informação que podia estar na RAM. A estrutura seguinte é a que contém os estado do jogo e é mantida na FPGA por ser pequena.

```
struct {
  unsigned 10 mcount, tcount;
   /* contadores: minas, tempo. */
  unsigned 4 mcount_digits[3];
   /* digitos vermelhos: minas. */
  unsigned 4 tcount_digits[3];
   /* digitos vermelhos: tempo. */
  unsigned 2 state;  /* estado do jogo.   */
  unsigned 3 smiley; /* índice do smiley. */
  unsigned 1 inside;
   /* cursor sobre as células e janela activa?*/
  unsigned 4 press_i, press_j;
   /* coordenadas da célula pressionada.      */
} game;
```

Durante o processo de limpeza automática recursiva de uma área de células sem minas, foi necessário implementar um stack em RAM, no qual são guardadas as coordenadas das células que precisam ser analisadas em cada ciclo. A variável `stack_p` foi inicializada com o último endereço da RAM de dados.

```
static void push(unsigned DataWidth val)
{
  RC100WriteSSRAM1(stack_p, val);
  stack_p--;
}
static void pop(unsigned DataWidth *val)
{
  stack_p++;
  RC100ReadSSRAM1(stack_p, *val);
}
```

Os bitmaps e o cursor foram obtidos a partir de ficheiros de imagens GIF e JPEG e convertidos para os formatos usados no programa usando um programa muito simples construído de propósito. Esse programa também gera as directivas `#define` com as características dos bitmaps. A sua descrição está fora do âmbito deste artigo mas é disponibilizado juntamente com este trabalho [3].

D. Estratégias gerais de optimização

Para optimizar o programa, nomeadamente, minimizar a ocupação da FPGA, usou-se, sempre que possível, a SSRAM1. Não podendo usar aquela RAM, declararam-se as variáveis como sendo dos tipos RAM e ROM. De notar que a FPGA é, ao mesmo tempo, o recurso mais importante e o mais limitado da placa RC100.

Também foram colocadas em prática as principais recomendações da Celoxica. Por exemplo, foram utilizadas expressões Handel-C partilhadas por oposição às expressões macro; foi completamente eliminado o uso de divisões; substituíram-se todos os ciclos `for` por ciclos `while`; substituíram-se as expressões aritméticas complexas por outras mais simples a correr em paralelo; eliminaram-se os *combinational cycles*, ainda que o acesso à RAM de vídeo pareça ter um na função `writePixel`; todas as expressões condicionais foram transformadas em operações entre valores de 1 bit; não se utilizaram dispositivos e periféricos desnecessários, por exemplo, o teclado, sendo o rato suficiente. Adicionalmente, isolou-se o envio de informação para a saída VGA da geração dessa informação, o que eliminou a necessidade de manter a escrita de píxels sincronizada com o *hardware*. O procedimento macro `runScreenAndMouse` podia ser mais optimizado, pois inclui cadeias `if-else` muito longas.

IV. CONCLUSÕES

Embora tendo como ponto de partida o simples exemplo "Mouse" incluído na documentação da ferramenta DK2, este trabalho constitui um exemplo bastante completo de como implementar um sistema gráfico de alto nível na placa RC100, usando a nosso favor as características do *hardware*. Neste sentido, os objectivos propostos foram integralmente cumpridos. Pela maneira que foi desenvolvido este programa, será possível adaptar o ambiente de execução a qualquer tarefa que necessite de interacção com o utilizador. Por fim, há que reconhecer que uma desvantagem deste sistema é a elevada ocupação da FPGA, cerca de 40% apenas com o ambiente gráfico. No entanto, o utilizador terá uma experiência idêntica ao uso de um moderno computador.

AGRADECIMENTOS

Ao Prof. Valery Sklyarov (Universidade de Aveiro), na sua constante disponibilidade e valiosa orientação durante as aulas, deu um precioso estímulo para a publicação deste artigo.

REFERÊNCIAS

[1] URL: http://www.celoxica.com/

[2] Adrian Nye, "Xlib Programming Manual", O'Reilly & Associates, Inc, 1989.

[3] URL: http://sweet.ua.pt/~leonel/work_aulas/cr_minesweeper/

Using High-level Languages for Hardware Modeling and Implementation

Nelson Ferreira, Filipe Teixeira,
Nuno Lau, Arnaldo Oliveira, Orlando Moreira[1]

Abstract – **This paper describes the use of high-level languages in hardware modeling and implementation. The purpose of the article is to describe a methodology that can be used in the design of a new system. First we will describe the main phases of hardware design flow, namely: modeling, validation, synthesis, implementation, prototyping and testing. We will also give a brief overview of some high-level languages. Afterwards, we will propose a methodology, where a new system is designed using successively a subset of C++, SystemC and VHDL using some guidelines to provide a smooth transition between languages and levels of abstraction. We will present a case study where an UART has been designed using this methodology. We will report the advantages and disadvantages of each language. This methodology provided a clear refinement flow from a functional sequential model to a RTL synthesizable model, although it created some consistency problems. The UART was implemented together with a MIPS32 processor within a FPGA for prototyping and testing purposes.**

Resumo – **Este artigo descreve a utilização de linguagens de alto nível na modelação e implementação de hardware. O objectivo deste artigo é apresentar uma metodologia que pode ser usada no projecto de novos modelos de sistemas. Primeiro iremos descrever as principais fases no fluxo de projecto de hardware, nomeadamente: modelação, validação, síntese, implementação, prototipagem e teste. Também iremos apresentar uma breve descrição de algumas linguagem de alto nível. Posteriormente, iremos propor uma metodologia usando algumas regras que permitem obter uma transição suave entre diferentes linguagens e níveis de abstracção, quando um sistema é modelado usando sequencialmente um subconjunto das linguagens C++, SystemC e VHDL nas diferentes fases do projecto. Será apresentado um *case study* do projecto de uma UART utilizando a metodologia proposta. Iremos expor as vantagens e desvantagens de cada linguagem. Esta metodologia permitiu obter uma passagem suave do modelo funcional até ao modelo RTL sintetizável, no entanto criou alguns problemas de inconsistência. A UART foi implementada para teste e propotipagem conjuntamente com um processador MIPS32.**

Keywords – **System Specification, Hardware Design Flow, Modeling, Synthesis, VHDL, SystemC, FPGA Prototyping, UART Design**

Palavras chave – **Especificação de um sistema, Fluxo de projecto de hardware, Modelação, Síntese, VHDL, SystemC, Prototipagem em FPGA, Projecto de uma UART**

I. Introduction

The first design stage of a digital system is the specification of its global functionality, the definition of the main interfaces and all other relevant characteristics and constraints. When designing a trivial system, natural languages are often used to build the respective specification. However, with the increase of system complexity, formal specifications are preferred because they can be verified, analyzed, simulated and synthesized with Computer Aided Design (CAD) tools.

Ideally, for complex systems, the specification must be the first step of a well defined design methodology. Models can be produced in a variety of high-level languages, such as C based languages or Hardware Description Languages (HDLs). Software engineers prefer software programming languages that provide a great level of abstraction. On the other hand, hardware engineers prefer HDLs, that provide adequate abstractions for hardware modeling. Furthermore when designing a system there is always the question of what should be implemented in software and what should be implemented in hardware. The ideal would be a tool that could, from a description in system level modeling language, separate what to implement in software and what to implement in hardware. Currently such tools are relatively immature, not widely available and mainly application domain specific [1].

After creating a model the developer must consider the validation, the synthesis, the implementation, the prototyping and finally the tests.

There are many methods to specify a digital system, namely boolean equations, schematic diagrams, graphical languages, HDLs, and system level modeling languages, depending on the abstraction level. We propose in this paper a methodology that starts by modeling the system in a C++ subset, and refines the model using SystemC [2] and VHDL [3]. This work has the following objectives:

[1]Philips Research Laboratories - Eindhoven

- Understand the suitability of different languages to each design phase.
- Establish the guidelines required to provide a smooth transition between different phases in the design flow using different languages.
- Create a methodology that implements those guidelines.
- Evaluate the methodology using a real world example.

This paper contains four more sections. In section II we explain the major steps in hardware development detailing the languages and the requirements of each one. In section III we present the proposed methodology. Section IV presents the development of an UART as a case study of the proposed methodology and present the results and the discussion. Finally, in section V we draw the conclusions.

II. HARDWARE DEVELOPMENT

A. Modeling

The result of specification is a model, i.e., a representation that shows the relevant characteristics without the associated details. It must incorporate all functional characteristics of the system without considering any implementation details such as specific components used or particular hardware/software partitions of the system implementation. Functional models are often used, at early design stages, to validate the algorithm that is going to be used in the system. The models can be produced in a variety of high-level languages, but in this case we used languages based on C and HDLs. In the section II-A.1 we describe some of these languages.

A.1 High-level Languages

C based languages

The C++ language is used in complex systems to write an executable specification and to develop software. The great advantage of the use of C++ for hardware modeling comes from its wide adoption and large programmers base. However, C++, in its original form, has some limitations to be used for hardware specification namely:

- Lack of appropriate data types;
- Absence of concurrency, reactivity and notion time;
- Great degree of freedom.

These can be considered the disadvantages of the use of C++ for hardware specification.

The SystemC [2] language was created to overcome the limitations of C++ in hardware modeling. SystemC is a set of class libraries implemented on top of C++, that supports hardware modeling concepts like concurrency, reactivity and latency. SystemC provides constructs that describe concepts that are familiar to hardware designers such as signals, modules and ports. In other words with SystemC we can define hardware and software components. SystemC also provides a simulation kernel that allows the designer to simulate the executable specification using just an ordinary C++ compiler. With SystemC the step-by-step refinement of a system design down to the RTL for synthesis is simplified.

Hardware Description Languages

VHDL [3] is a hardware description language where we can describe the model of an hardware system. The goal of VHDL creators was to create an unambiguous, portable and general language. However, because VHDL is a strongly typed language, the syntax of VHDL becomes verbose (e.g. additional code is often needed to explicitly convert one data type to another). On the other hand, the strongly type feature can be beneficial to detect errors at early design stages.

B. Validation

The validation of a model can be made in two different ways: formal verification and validation through simulation. In the design of a system the validation must be executed at several stages in order to validate the results obtained in the stages that precede it and to detect errors as soon as possible.

The validation through simulation is the most used when it is intended to make a validation of a system. The objective is to execute a logical verification and make an analysis of the performance. Generally, the use of testbenches is convenient to execute a logical verification. They apply stimulus to the system inputs. This type of verification has the disadvantage of being insufficient for complex systems, because it isn't an exhaustive method. In most complex projects it is impossible for test vectors to cover all the cases.

The formal verification is based on the use of mathematical methods to verify the functionality of the system. It has the advantage of not needing test vectors and supplies an exhausting verification. It can be used as a complement of the previous method.

C. Synthesis and Implementation

The synthesis is the process used to obtain, from the behaviorial description, a structural description in a lower abstraction level. Generally the synthesis is the partition in modules of the behavioral description. Depending on the abstraction level we can have different kinds of synthesis, such as system synthesis and logic synthesis. With system synthesis it is possible to decompose an abstract specification of the system in a software implementation and a hardware implementation. One objective of the system synthesis can be the reuse of predefined components, e.g., *Intellectual Property* blocks. Logic synthesis generates the circuit that implements a given logic specification, e.g., generate circuits with finite state machines through connection of flip-flops. Synthesis can be realized manually, or automatically using CAD tools, although the manual

process is slow and error prone. The final result of synthesis is a circuit described in a the form of a netlist.

Finally, the implementation is the process of mapping, placement and routing. Mapping is the adaptation of our netlist to the components available in a given technology. Placement is the positioning of each component in the available area of the implementation. Routing determines the path of the signals that connect the component interfaces.

D. Prototyping and Testing

After implementation of the design the developer comes across with the necessity to foresee the real behavior of its design.

Prototyping consists in the creation of a functioning version of the final system without some of the deployment characteristics. Some constraints related to area, cost and power consumption may be relaxed in this phase.

III. Proposed Methodology

Our methodology proposes the guidelines required to provide a smooth transition from behavioral abstraction level to RTL synthetisable level using different languages at different stages. It uses C++ to create the executable specification, then SystemC to create the hardware/software models and finally VHDL to obtain a synthetisable model. C++ was chosen due to its encapsulation capabilities and because it is the SystemC base language. In figure 1 we present a diagram with the transition between the different stages.

We start by defining the specification of the system, i.e, the interfaces and the behavior. To guarantee a smooth transition, we propose some coding guidelines, that must be taken into account while building the functional model:

- Each component must be implemented as a C++ class.
- Use few data types and avoid the use of pointers.
- Hardware ports are modeled as class constructor parameters and port bindings are implemented using external shared variables.
- Asynchronous and synchronous events are implemented as independent functions.
- Because of the lack of concurrency support, the order of function invocation must be considered, for a correct simulation progress.

If these guidelines are followed the transition to SystemC is smooth and with very little changes from the C++ model. The functions used in the C++ model to simulate synchronous and asynchronous events are translated to signals of SystemC method sensitivity list. The use of HDLs at the final stage of development is justified by the fact that the synthesis tools are more developed for these languages, the synthetisable language subset is well documented and for its strong checking capabilities.

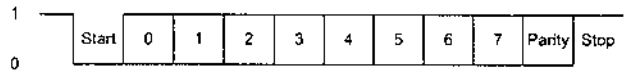


Figure 2 - Frame format

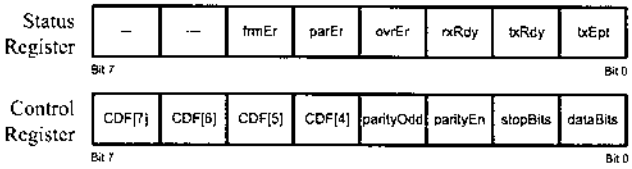


Figure 3 - UART Control and Status Registers

The innovation of this methodology isn't the successive use of C++, SystemC and VHDL at the different design stages [4] [5] but it is the application of the guidelines that provide a smooth transition from behavioral abstraction level to hardware synthesis.

IV. A Real World Example

A. The RS232 Protocol

The RS232 Protocol is an asynchronous serial communication [6] method used in point to point interfaces. The protocol describes a communication method where transactions of information are made character by character. In an asynchronous communication link there is no separate clock line, thus the data must be synchronized using others methods, such as special synchronization bits within the data frame.

Another important consideration is the transmission baud rate. The transmitter and receiver must be programmed to use the same bit frequency.

Figure 2 shows the frame in RS232 Protocol composed by a *Start Bit*, *Data Bits*, *Parity Bit* and *Stop Bits*. The *Start Bit* is used for synchronization purposes. Because the line is in marking state (on state) when idle, the *Start Bit* (off state) is easily recognized by the receiver. The *Data Bits* are sent immediately following the *Start Bit* and the least significant bit is always the first bit sent. The *Parity Bit* exists for detecting errors. The parity can be calculated in Even parity or in Odd parity. The *Stop Bits* identify the end of a data frame, in other words we can say that the stop bit is the minimal interval between two consecutive characters.

B. UART Interface

An UART is a hardware component used to implement RS232 protocol. Our implementation provides a synchronous memory-like bus interface for applications that communicate over RS232. The UART has a control register where the transmission parameters can be defined and a status register where some flags are stored. Besides the control and the status register the UART also has two data registers: a transmission buffer and a reception buffer. All registers are eight bits wide. The status register (see figure 3) has information about the status of transmission and reception buffers and also if errors have been detected. As we can see in figure 3 the status register is composed by:

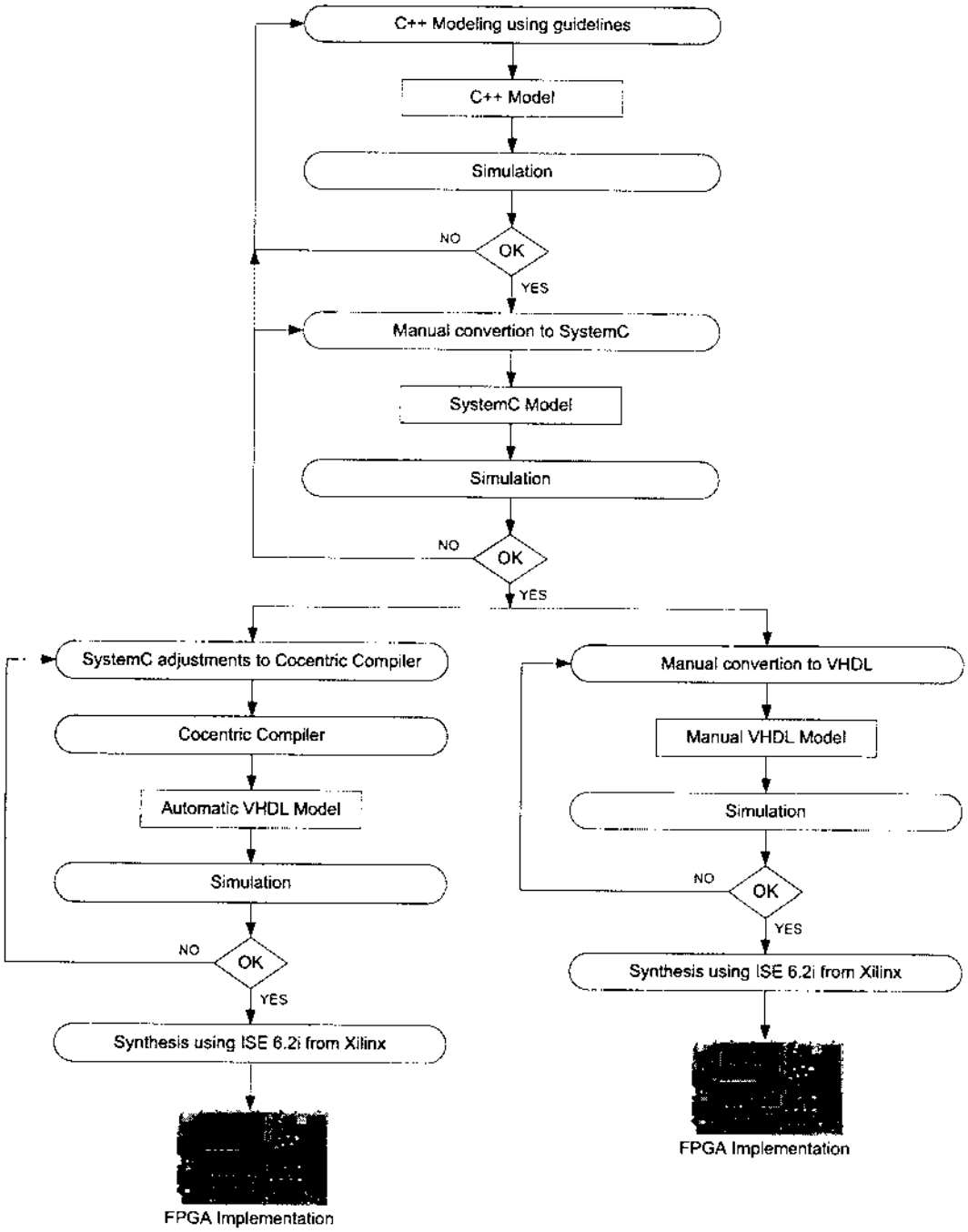


Figure 1 - Methodology diagram

frmEr – is set to "1" to indicate an invalid *Stop Bits* field due to noise, synchronization errors or configuration mismatches.

parEr – this bit is set to "1" to indicate that there is a parity error.

ovrEr – this bit is set to "1" when new data has been received and the previous data has not been read from reception buffer.

rxRdy – this bit is set to "1" to indicate that reception buffer has received a new character.

txRdy – this bit is set to "1" to indicate that the transmission buffer is ready to accept a new character.

txEpt – this bit is set to "1" to indicate that there is no character in the transmitting shift register.

The control register (see figure 3) is a programable register where the user can define some parameters such as:

CDF – these 4 bits define the value of the clock divider.

parityOdd – define the parity (odd or even).

parityEn – determines the use (or not) of a parity bit.

stopBits – defines the number of stop bits. The bit set to "1" indicates 2 stop bits, otherwise 1 stop bit is used.

dataBits – defines the number of data bits to transmit. The bit set to "1" indicates 8 data bits, otherwise

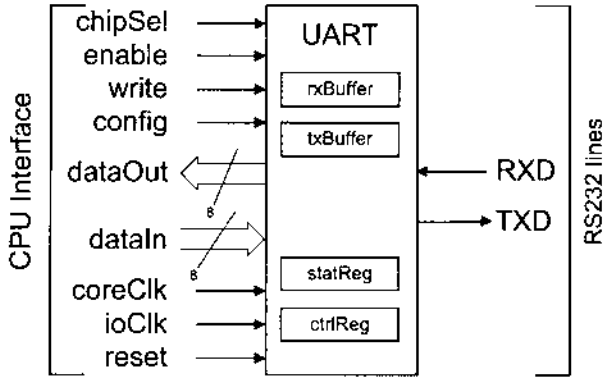


Figure 4 - UART Schematic

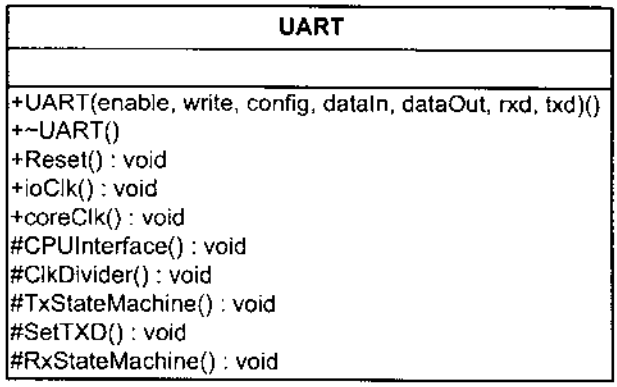


Figure 5 - CLASS UART

7 data bits are used.

In figure 4, we can see the ports that provide the interface with CPU, and the interface with the RS232 line. The *dataIn* and *dataOut* are eight bits wide buses, the remaining ports are used for control purposes. The values at each port are shown in table I. The interface with protocol is made with the lines *RXD* and *TXD*.

C. UART Specification

Before starting to build any model we have to define precisely what we pretend to obtain in the end. With this in mind we defined the specifications of our model, these specifications are done without the details of the implementation. We only describe what we pretend to obtain.

With this in mind we chosed to have two clock signals, one clock implements the synchronous interface between the CPU and the UART (*ioClk*), the second one is used to obtain the baud rates necessary for transmission and reception using a clock divider (*coreClk*).

To set the baud rates we use the four MSB of control register. In practise we can define values between 16 and 192 for clock divider then the baud rates varied between 9600 and 115200 for a frequency of 1.843Mhz to *coreClk*.

The UART must be able to transmit and receive at the same time, i.e., full duplex communication must be provided.

C.1 C++

C.2 Modeling

The modeling in C++ consisted in the construction of a class (figure 5) that allowed us to obtain a functional model of the UART.

The data types used to specify the variables were `bool` for single bit variables and `unsigned char` for eight bit variables. Regarding interface ports access, the shared variables that model the ports are passed as arguments to the class constructors (figure 5). The same shared variable can be used to connect multiple components, e.g., these variables may be shared between the CPU and the UART or between two UARTs. In

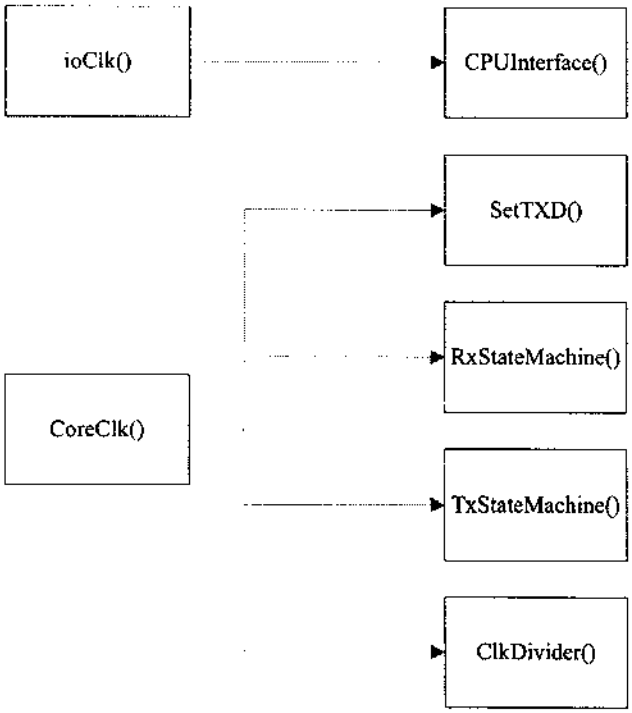


Figure 6 - Functions dependency

the definition of the constructor we declare the parameters that are used to simulate the input ports of the UART interface as `const`, this way functions inside the class cannot set values to these parameters. Because C++ doesn't support the notion of time, we had to implement two functions, `ioClk()` and `coreClk()`, that simulate the synchronous events generated by the clocks. In hardware the functions that are triggered by an event are executed in parallel and in C++ they are executed sequentially, hence we had to take some cautions in the order of invoking the functions that implement the functionality of the design.

The functions that implement both clocks where subdivided in others functions as shown in figure 6, these functions are responsible for the implementation of the UART internal operations.

Function `CPUInterface()` implements the synchronous communication between the CPU and the UART.

	Operation						
Signals	Initialization	No Operation		WR Ctrl Reg	RD Stat Reg	WR Tx Buf	RD Rx Buf
reset	1	0	0	0	0	0	0
chipSel	x	0	x	1	1	1	1
enable	x	x	0	1	1	1	1
config	x	x	x	1	1	0	0
write	x	x	x	1	0	1	0
dataIn	xxxxxxxx	xxxxxxxx	xxxxxxxx	ctrl reg val	xxxxxxxx	char to tx	xxxxxxxx
dataOut	zzzzzzzz	zzzzzzzz	zzzzzzzz	zzzzzzzz	stat reg val	zzzzzzzz	rx char

Table I
PORT VALUES FOR EACH UART INTERFACE OPERATION

```
void UART::CPUInterface() {
    if ((m_enable == 1) && (m_chipSel == 1)) {
        if (m_config == 0) {
            if (m_write == 1){
                m_txBuffer = m_dataIn;
                m_statReg.m_bits.txRdy = 0;
            } else {
                m_dataOut = m_rxBuffer;
                m_statReg.m_bits.rxRdy = 0;
            }
        } else {
            if (m_write == 1) {
                m_ctrlReg.m_byte = m_dataIn;
            } else {
                m_dataOut = m_statReg.m_byte;
                m_statReg.m_bits.ovrEr = 0;
                m_statReg.m_bits.parEr = 0;
                m_statReg.m_bits.frmEr = 0;
            }
        }
    }
}
```

Figure 7 - C++ implementation of the function CPUInterface()

To provide the full duplex capability, we had to have two separate modules, one for transmission and one for reception. The module that implements the transmission is composed by two functions. `TxStateMachine()`, represents the transmitting state machine. The function `SetTxd()`, implements a multiplexer that, using the information provided by `TxStateMachine()`, drives the *TXD* line.

The module that implements the reception is composed by one function, `RxStateMachine()`, this function represents a state machine that is in charge with the reception of a character.

It is also necessary to implement a clock divider. Function `ClkDivider()` generates the standard baud rates that are used in the protocol communication.

The asynchronous reset interface is modeled with the independent function `Reset()`, this function initializes the internal variables of the UART.

To demonstrate the different implementations and changes in the languages used in our case study we will use the function `CPUInterface()`. In the C++ design the function has the code showed in figure 7.

This function implements a synchronous interface between the CPU and the UART, four operations can be performed by the CPU on the UART:

- Write to the control register
- Write to the transmission buffer
- Read from the status register
- Read from the reception buffer

The signals *enable* and *chipSel*, activate the CPU interface. When they are active, the signals *write* and *config* are tested to select the operation to perform. Depending on the operation, this function also performs the reset of some status bits.

C.3 Simulation

The simulation in C++ was based on the communication between two UARTs (figure 8). UART1 is responsible for transmitting the character pressed on the keyboard to UART2 and UART2 is responsible for the reception of the character and for sending it to the display. With this simple simulation we test both transmission and reception of the UART. To implement this simulation we created a `main` function that performs the following tasks: writes into the UART1 transmission buffer all characters pressed in the keyboard, polls the UART2 to detect new received characters and sends them to the display. The UART interface signals are driven accordingly to the operation performed.

We also created a class that manipulates Value Change Dump (VCD) files. This class was responsible to create and maintain one VCD file that stores the evolution in time of variables values. To visualize these files we used the application GTKWave [7]. This simulation allowed us to test the external interface of the UART, using the functional testbench, and the internal details, using the VCD file.

D. SystemC

D.1 Modeling

The transition from C++ to SystemC was done with very little changes in terms of the code to implement the functions. Changes occurred mainly in the declaration of the functions and the type of variables. In

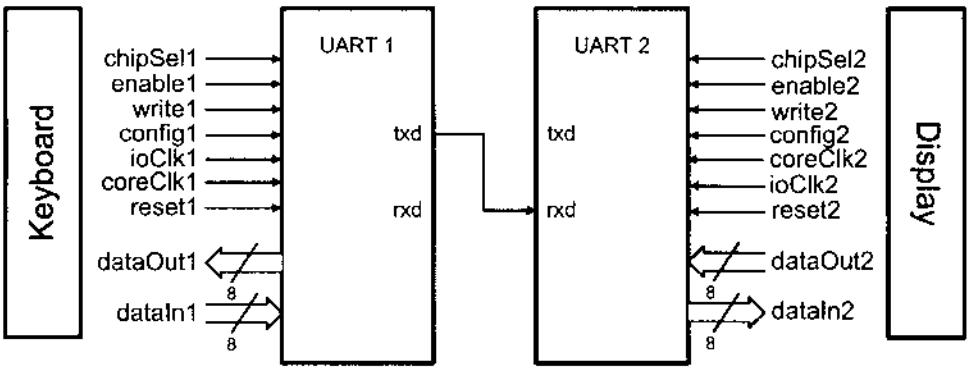


Figure 8 - Simulation setup

SystemC the UART class became a `SC_MODULE`, and some of the functions were implemented as `SC_METHODs` with a sensitive list of the signals that can trigger the method.

Unlike C++, SystemC provides concurrency and reactive behavior, so the functions that implemented the clocks, and the asynchronous events are no longer necessary. Event handling is transparently managed by the SystemC simulation kernel using the sensitivity list of the methods. Hence, in the SystemC model we don't have the function `Reset()`, all methods are sensitive to the signal *reset* and they perform the reset to the signals that they are responsible to drive. The addition of the reset capability to the former C++ functions implied some modifications in the code.

All the variables were converted to SystemC data types, variables declared as `unsigned char` were declared as `sc_lv<8>` or as `sc_uint`. All variables were declared as `sc_signal`.

The code showed in figure 9 implements the CPUInterface method using SystemC:

The modifications from C++, are mainly related with the integration of the reset functionality in the model. The other modifications are only related with the access to the SystemC data types.

```
void  UART::CPUInterface() {
 if (reset == 1) {
    s_ctrlReg = 0xF0;
 } else {
    if ( (ioClk.event())
         && (ioClk.read() == 1)) {
       if ((enable.read() == 1)
           && (chipSel.read() == 1)) {
          if (config.read() == 0) {
             if (write.read() == 1) {
                 s_txBuffer = dataIn.read();
                 s_statReg[1] = 0; }
             else {
                 dataOut.write(s_rxBuffer);
                 s_statReg[2] = 0; }
          } else {
            if (write.read() == 1) {
                s_ctrlReg = dataIn.read();
                sc_lv<8> aux = dataIn.read().range(7,4);
                s_clkDivFactor.range(7,4) = aux; }
             else {
                dataOut.write(s_statReg);
                s_statReg = (s_statReg & 0x07); }
          }
       }
    }
 }
}
```

Figure 9 - SystemC implementation of the function CPUInterface()

D.2 Simulation

The simulation in SystemC was done with a testbench. To implement it we created a new module. The testbench module was specified with a process of type `SC_THREAD`. In that process we provided the values that must be assigned to the interface of the UART in a sequential form and after each modification we add the function `wait()`.

Functions to manipulate VCD files are already supplied by the SystemC library. To visualize the files produced we used the application GTKWave.

D.3 Synthesis and Implementation

To perform the SystemC synthesis we used the aplication CoCentric SystemC Compiler [8].

Although the code produced so far simulated correctly, to be able to convert it to VHDL using the CoCentric compiler we had to make some changes because the application has the following restrictions:

- Does not support multi-source signals, i.e., although signals can be read by multiple methods they can only be changed by one method.
- For single bit variables the use of the data type `bool` is advised.
- Does not support the function `event()` inside the methods to test witch signal generated the event.
- Sensitivity lists that mix level and edge sensitivities are not allowed.

The signal that was causing multi-source problems was the status register, because we had to change its bits in the methods used for the state machines and in the method that implemented `CPUInterface()`. The bits that were causing multi-source problems were the *txRdy* because it is set in the method `TxStateMachine()` and is reset in the

	VHDL by Cocentric	Manual VHDL
Max. Frequency	55.408Mhz	55.408Mhz
# slices used	152 out of 3072	122 out of 3072
# slice Flip Flops	112 out of 6144	102 out of 6144

Table II
DEVICE (SPARTAN II-XC2S300E) UTILIZATION OBTAINED USING SYNTHESIS TOOLS

method `CPUInterface()`, the bit *rxRdy* because it is set in the method `RxStateMachine()` and is reset in `CPUInterface()`, and the bits *ovrEr*, *frmEr* and *parEr* because they are set in the method `RxStateMachine()` and the reset in `CPUInterface()`.

The solution adopted consisted in creating additional methods that generate a reset signal that is added to the sensitive list of the methods `RxStateMachine()` and `TxStateMachine()`, this way only the later methods change the bits from the Status Register.

After resolving these problems we were able to obtain a VHDL code, the code was synthesized using the application Xilinx ISE 6 and the Spartan II-E FPGA as target device. The results are presented in table II.

E. VHDL

In order to assess the quality of the VHDL code produced by the CoCentric SystemC Compiler, we performed the manual translation of the SystemC code into VHDL. The hand coded VHDL was then synthesized and the results were compared. This translation also allowed us to evaluate the involved effort, validating the proposed methodology.

E.1 Modeling

The manual conversion to VHDL consisted in transforming the SystemC methods into processes, convert the data types to VHDL data types and make the necessary changes in the syntax of the languages.

The VHDL code that implements the `CPUInterface` process is showed in figure 10.

E.2 Simulation

The simulation in VHDL was made using ModelSim® with a testbench that is similar to the one implemented in SystemC with correct adjustments to the VHDL language.

E.3 Synthesis and Implementation

The VHDL module was synthesized with the application Xilinx ISE 6 [9] to be used in the device Spartan II-E FPGA, the synthesis generated the device utilization summary presented in table II.

F. Comparing Methods

F.1 Modeling

Comparing the three methods we used we can conclude that C++ is certainly adequate for modeling the

```
cpu_interface : process(reset, ioClk) begin
    if (reset = '1') then
        s_ctrlReg <= "11110000";
    elsif (rising_edge(ioClk)) then
        if ((chipSel = '1') and (enable = '1')) then
            if (config = '0') then
                if (write = '1') then
                    s_txBuffer  <= dataIn;
                else
                    s_dataOut   <= s_rxBuffer;
                end if;

            else
                if (write = '1') then
                    s_ctrlReg   <= dataIn;
                else
                    s_dataOut   <= s_statReg;
                end if;
            end if;
        end if;
    end if;
end process;
```

Figure 10 - VHDL implementation of the function CPUInterface()

software components of a system and to test the algorithm, but when it comes to modeling the hardware behavior it lacks some functionality.

SystemC, even if it is based on C++, has some functionalities that are more hardware like, and it benefits from a higher level of abstraction than conventional VHDL design.

The VHDL model is the most close to the hardware specification, however we think that the constraints imposed by the hardware description languages would be inconvenient in the early development stages. In our opinion the quality of the final VHDL code resulting from the proposed methodology beneficiated from the iterative refinement process.

F.2 Simulation

The simulation in C++ is more efficient at early stages of development to validate the algorithms and the sequence of operations without worrying about clocks and precise timings. It also helps in the functional decomposition of the system.

SystemC provides the concepts of concurrency and parallel execution of processes. Using SystemC modules run conceptually in parallel which may allow the detection of hidden data dependencies that were not visible in the C++ simulation. SystemC directly supports the generation of VCD files, which facilitates the visualization of the simulation results.

Two models of the same circuit, one written in SystemC and the other in VHDL must produce the same simulation output. However, the simulation of the SystemC model can be done with the SystemC library and a standard C++ compiler, which are both freely available. On the other hand the simulation in VHDL requires a specific simulator.

`void UARTSetup (TBaudRate baudRate, TDataBits dataBits, TParity parity, TStopBits stopBits);` This function programs the transmission parameters of the UART.
`void UARTTxChr (char chr);` This function sends a character to the UART to be transmitted.
`bool UARTRxChr (char* chr);` This function reads a received character from the UART.
`void UARTTxStr (const char* str);` This functions sends a string to the UART to be transmitted.

Table III
UART API DESCRIPTION

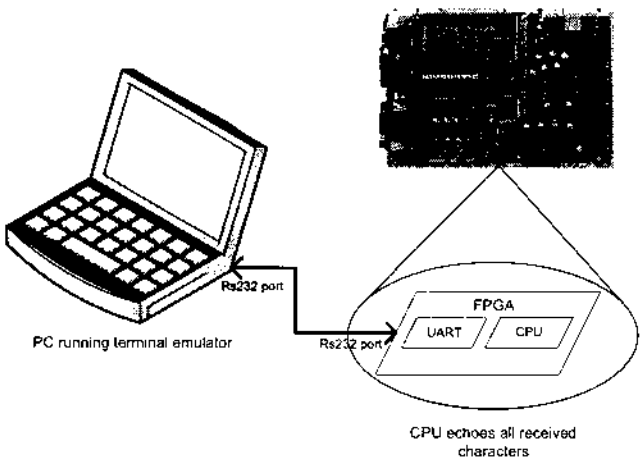


Figure 11 - Test setup

F.3 Synthesis and Implementation

The synthesis of the UART was made in two different ways: directly from the SystemC code using the Cocentric SystemC compiler and from hand-coded VHDL model using the Xilinx XST synthesis engine incorporated within Xilinx ISE Design environment.

The Cocentric SystemC compiler constraints force a non natural coding style. To obtain the SystemC synthesizable code we had to analyze how different SystemC constructions were synthesized in order to make the compiler generate synthesis friendly VHDL code.

The results of both synthesis flows are summarized into table II. Comparing results we can conclude that direct VHDL synthesis provides better results in terms of area.

G. Testing

The test of the UART was made in a FPGA (Development Board model TE-XC2SE from Trenz [10]) integrated with a MIPS32 processor [11].

The test consisted in the simulation of the communication between two computers. The program implemented in the MIPS32 just echoes all the characters received by the UART back to the computer. Using a terminal emulator we established a communication between the computer and the development board as seen in figure 11.

G.1 The UART API

To test the UART with the MIPS32 processor, we had to create an interface library. The library is described in table III.

V. CONCLUSION

The discussion on the utilization of C++, SystemC and VHDL at different stages of development presents a clear view of the advantages and disadvantages of each language.

The proposed methodology does not solve the consistency problems when translating the model manually. We also came across with the lack of support for this methodology of some of the tools that were used.

With our example we have shown that it is possible to have a smooth transition from the various phases of designing a system, using different design languages, when we follow the proposed guidelines.

REFERENCES

[1] Wayne Wolf, "A decade of hardware/software codesign", *IEEE Computer*, vol. 36, no. 4, pp. 38–43, April 2003.

[2] *Open SystemC Initiative*, http://www.systemc.org.

[3] IEEE, *IEEE Standard VHDL Language Reference Manual*, 2000 edition.

[4] Eike Grimpe and Frank Oppenheimer, "Extending the systemc synthesis subset by object-oriented features", in *Proceedings of the 1st IEEE/ACM/IFIP international conference on Hardware/software codesign and system synthesis*, 2003, pp. 25–30, ACM Press.

[5] Yuval Ronen J.R. Armstrong, "Modeling with systemc: A case study", 2001.

[6] Electronic Industries Association, *EIA232E - Interface Between Data Terminal Equipment and Data Circuit-Terminating Equipment Employing Serial Binary Data Interchange*, 1991.

[7] *GTKWave Homepage*, http://www.linux-workshop.com/bybell/ver/wave.

[8] *Describing Synthesizable RTL in SystemC, Version 1.2, November 2002*, http://www.synopsys.com.

[9] *Xilinx, inc.* http://www.xilinx.com.

[10] Trenz Electronic, http://www.trenz-electronic.de, *Spartan-IIE Development Platform Overview*, 2004.

[11] Arnaldo S. R. Oliveira António B. Ferrari , Valery A. Sklyarov, "Arpa - an open source system-on-chip for real-time applications", ERTSI - Embedded Real-Time Systems Implementation Workshop, 2004.

Interacção com um Cubo de Rubik Virtual

Carlos Silva, Milton Ruas

Abstract – **There are several ways to manipulate a virtual Rubik's Cube, either using a large number of keys from a PC's keyboard (inefficient for large cubes) or bringing the mouse into action. In this paper we present an efficient method to manipulate the cube using the mouse (or mouse and keyboard).**

Resumo – **Há muitas formas de manipular um Cubo de Rubik virtual, desde métodos que impliquem o uso de grande parte das teclas disponíveis no teclado (bastante ineficiente para cubos grandes) a métodos que envolvam também o uso do rato. Neste artigo apresentamos uma forma eficiente de manipulação do cubo com o rato (isolado ou em combinação com o teclado).**

I. HISTÓRIA

O *Cubo de Rubik* nasceu na Hungria em 1974 através de Erno Rubik, cuja paixão pela arte e pela técnica o levou à criação de um jogo em que o oponente é a própria natureza. A curiosidade pelo espaço e a relação deste com o Homem, que está na base da sua formação em Arquitectura e Design, levou-o a pensar na figura geométrica do cubo [1]. Desde a sua origem até hoje, este jogo tem atraído inúmeras pessoas, existindo alguns campeonatos por todo o mundo. Actualmente existem várias versões do Cubo de Rubik destinadas aos apaixonados deste objecto, das quais se salientam o cubo de $5 \times 5 \times 5$ e o cubo 4-D.

Existem vários métodos para resolução do Cubo de Rubik, todos eles resultando da Teoria de Grupos [2]. Esta foca o estudo da simetria de forma abstracta e fornece uma ligação entre o espaço e a estrutura. As aplicações desta área da Matemática são vastas: Cristalografia, Mecânica Quântica, Física Molecular, etc.

O Cubo de Rubik é, desta forma, uma aplicação que permite desenvolver as capacidades intelectuais e de discernimento espacial a quem se propõe resolvê-lo.

II. INTRODUÇÃO

O Cubo de Rubik é um cubo definido por uma cor diferente em cada uma das faces (quando resolvido), sendo constituído por vários cubos mais pequenos. Os cubos pequenos não são fixos, podendo deslocar-se com alguma independência em relação aos outros, mas nunca individualmente. O deslocamento é feito pela rotação de uma linha ou coluna de cubos (discos cúbicos) em relação a qualquer um dos eixos tridimensionais. O objectivo do jogo é, pela manipulação dos discos, obter uma só cor em cada face do Cubo de Rubik.

No âmbito da disciplina de Computação Gráfica (LEET, opção de 5º ano), pretendia-se construir uma aplicação que simulasse o tradicional Cubo de Rubik para vários tamanhos diferentes (de $2 \times 2 \times 2$ a $10 \times 10 \times 10$), usando as bibliotecas OpenGL e GLUT [3].

Como o cubo tradicional é essencialmente jogado com as mãos, foi necessário fornecer um método suficientemente inteligente que permitisse que a interacção com o computador fosse o mais intuitiva possível, tornando assim, como veremos, o uso do rato imprescindível.

III. MANIPULAÇÃO GLOBAL DO CUBO

A manipulação do modelo, para permitir a visualização de todas as faces do cubo, é feita de modo a permitir rodar o cubo em torno de qualquer eixo tridimensional.

Estão disponíveis três formas de o fazer:

- Rotações incrementais utilizando o teclado;
- Rotações automáticas de $\pm 90°$, segundo qualquer eixo, utilizando o teclado;
- Rotações automáticas de $\pm 90°$ utilizando o rato.

As rotações automáticas usando o rato são executadas premindo o botão esquerdo com o cursor no exterior do modelo, fazendo com que, dependendo do local, o modelo rode no sentido apropriado. Foram estabelecidas quatro zonas da janela para efectuar estes movimentos (figura 1).

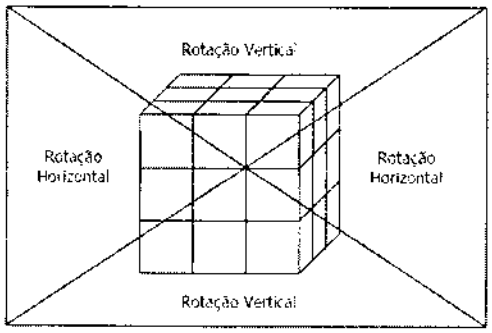


Figura 1 - Zonas para a rotação do modelo usando o rato

A identificação da posição do cursor é feita por verificação das coordenadas do pixel seleccionado relativamente às duas rectas fronteira (verificando se o pixel está acima ou abaixo de cada uma das rectas é possível determinar qual a zona seleccionada).

IV. MANIPULAÇÃO DOS DISCOS CÚBICOS

O Cubo de Rubik, como já foi referido, é constituído por um conjunto de cubos mais pequenos de características próprias. Por isso, na representação interna do modelo do Cubo de Rubik, cada um desses cubos possui um descritor do seu estado, na forma de uma estrutura com os seguintes campos:

Parâmetros de translação local: que indicam a posição relativa do cubo no modelo (figura 3);

Parâmetros de rotação local: que permitem a rotação dos cubos, em torno do centro do modelo, de forma independente dos outros;

Propriedades: de cada uma das faces (componentes de cor).

As propriedades das faces são armazenadas num *array* de seis elementos (seis faces por cubo), cujas referências seguem o modelo da figura 2.

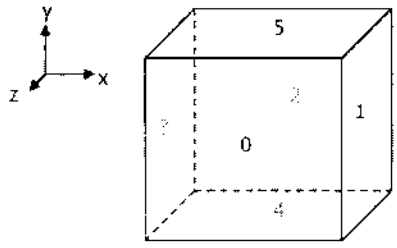


Figura 2 - Referências das faces de cada cubo

Os diversos cubos são organizados num *array* tridimensional de ponteiros que apontam para os descritores dos cubos. Tal escolha deveu-se à simplicidade na referência a um determinado cubo, usando apenas as coordenadas x, y, z (figura 3).

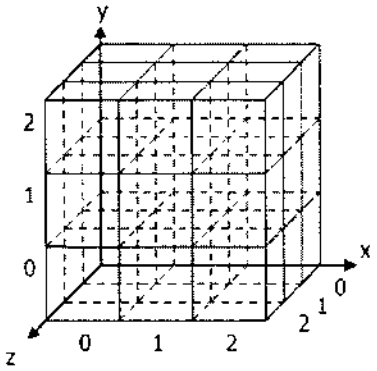


Figura 3 - Coordenadas dos cubos num modelo $3 \times 3 \times 3$

Para a rotação de um disco cúbico, é necessário rodar vários cubos ao mesmo tempo. Por exemplo, no modelo de $3 \times 3 \times 3$ é necessário rodar nove cubos simultaneamente segundo um ângulo de $\pm 90°$.

A primeira abordagem que adoptámos foi, simplesmente, aplicar uma transformação de rotação de $90°$ aos cubos pertencentes ao disco. Apesar de funcionar para o primeiro movimento, apresenta falhas nos seguintes: o deslocamento é feito por referência directa aos cubos a mover do *array* tridimensional, e considerando que a referência de um cubo corresponde à sua posição absoluta segundo a figura 3, logo após o primeiro movimento as referências dos ponteiros já não estão relacionados com as posições dos cubos, conduzindo este método ao fracasso.

De modo a corrigir este problema, decidiu-se evitar o uso de transformações geométricas, e elaborar um algoritmo que contemplasse a actualização dos ponteiros dos cubos. Este contempla quatro passos a executar para cada cubo a deslocar:

1. Cálculo da referência do cubo que irá substituir o actual. Para tal, são efectuadas as atribuições:
   - Para rotações de $+90°$
     $var1(novo) = rubik.ladocubos - 1 - var2(antigo)$
     $var2(novo) = var1(antigo)$
   - Para rotações de $-90°$
     $var1(novo) = var2(antigo)$
     $var2(novo) = rubik.ladocubos - 1 - var1(antigo)$

   $rubik.ladocubos$ indica o número de cubos que o modelo tem de lado (3 para o modelo $3 \times 3 \times 3$); $var1$ e $var2$, são as componentes x, y ou z das coordenadas dos cubos, e dependem do eixo de rotação (figura 3):

	Rotação XX	Rotação YY	Rotação ZZ
$var1$	Y	Z	X
$var2$	Z	X	Y

2. O ponteiro do cubo actual passará a apontar para o novo cubo, que ocupará a posição do cubo actual após o deslocamento;
3. Actualização da posição relativa do novo cubo (alteração dos parâmetros de translação local);
4. Rotação das propriedades das faces. É importante verificar que as faces terão de rodar segundo o mesmo eixo de rotação do cubo, para que as cores correspondam às posições certas depois do deslocamento (figura 2).

Um pormenor importante e que se deve ter em conta, é que a actualização dos ponteiros segue um caminho circular (ao longo do disco a mover), ou seja, ponteiros actualizados serão novamente utilizados para consulta na actualização de futuros cubos (as suas posições antigas passarão a ser as posições de outros cubos), o que representa um problema pois os ponteiros consultados deverão sempre apontar para as posições antigas.

Para resolver esta questão utilizou-se um *array* temporário de dimensões iguais ao primeiro, que inicialmente aponta para os mesmos descritores que o *array* principal. Este *array* será utilizado para consultar as posições antigas dos ponteiros, actualizando apenas o *array* principal.

Com este algoritmo, todos os deslocamentos são feitos com sucesso para cubos de qualquer dimensão.

V. INTERACÇÃO COM O MODELO

Três métodos de interacção com o modelo são possíveis, no que diz respeito à rotação dos discos:

- Usando apenas o teclado;
- Usando apenas o rato;
- Usando simultaneamente o teclado e rato.

A. Interacção usando apenas o teclado

Este é o método mais simples. Foram atribuídas duas teclas, correspondentes à rotação no sentido directo ou inverso, para cada disco de cubos. Usando o modelo de $2 \times 2 \times 2$, temos 6 discos deslocáveis, pelo que são precisas

12 teclas no total, para efectuar todo o tipo de movimentos (figura 4).

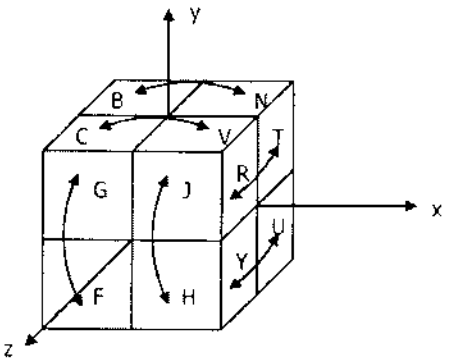


Figura 4 - Manipulação do modelo 2 × 2 × 2 usando apenas o teclado

Este princípio também funciona no modelo de 3 × 3 × 3, contudo apenas manipula os discos extremos. O disco central apenas pode ser deslocado por manipulação simultânea dos discos extremos.

Já a partir do modelo de 4 × 4 × 4, este método torna-se incompleto, pois existem discos que nunca poderão ser rodados, pelo que não foi utilizado a partir destas dimensões.

Apesar de ser um método simples, é pouco intuitivo, dada a elevada quantidade de teclas a usar. Leva também a algum tempo de adaptação por parte do jogador e, além disso, variando a orientação do modelo perde-se a noção das teclas.

B. Interacção usando apenas o rato

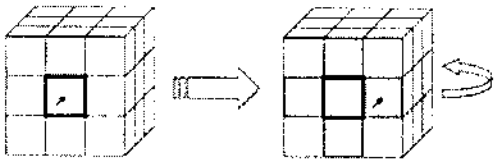


Figura 5 - Rotação dos discos usando apenas o rato

Procedimento:

1. Seleccionar uma face referência – um contorno colorido surge à volta desta face;
2. Seleccionar uma face ao lado, no sentido do deslocamento pretendido.

Este é o método mais simpático ao utilizador, dado que é muito intuitivo e não depende da orientação do modelo. No entanto, a implementação torna-se mais complexa, dado que é necessário relacionar o pixel seleccionado com o objecto desenhado.

Tal complexidade provém do facto de ser necessário percorrer o *pipeline* de visualização ao contrário, tarefa nada fácil de realizar, embora teoricamente possível. O que o OpenGL faz é, em vez de fazer o *rendering* das faces a seleccionar no *color buffer*, fá-lo no *select buffer*, definindo para cada uma delas um identificador. Deste modo, é só seguir as suas posições sempre que são desenhadas, e compará-las com a posição de selecção do rato [4] [5].

Sempre que a posição de uma ou mais faces coincide com a selecção do rato, várias mensagens (uma por cada face seleccionada) são geradas e armazenadas num *stack buffer*. Deste modo, cada mensagem deve ser analisada separadamente e apenas escolhida a face que se encontra mais próxima do utilizador.

A atribuição de um identificador a cada face é feita na forma de um número inteiro, e segue a seguinte expressão (assim cada face possui um identificador único diferente de todos as outros):

$$name = (x \times rubik.ladocubos^2 + y \times rubik.ladocubos + z) \times 6$$

Quando o rato selecciona uma face, a indicação do cubo e da face seleccionada em termos da posição (x, y, z, face) é obtida da seguinte forma:

$$x = \left\lfloor \frac{name}{rubik.ladocubos^2 \times 6} \right\rfloor$$

$$y = \left\lfloor \frac{name}{rubik.ladocubos \times 6} \right\rfloor \% \; rubik.ladocubos$$

$$z = \left\lfloor \frac{name}{6} \right\rfloor \% \; rubik.ladocubos$$

$$face = name \% 6 \qquad \text{(6 faces para cada cubo)}$$

Este método de selecção, apesar de ser bastante fácil de usar, apresenta algumas desvantagens. A principal é que, devido à elevada duração na resolução do Cubo de Rubik, o uso do rato torna-se cansativo.

Uma forma de tentar resolver este problema é o terceiro método de interacção: usando teclado e rato simultaneamente.

C. Interacção usando o teclado e o rato

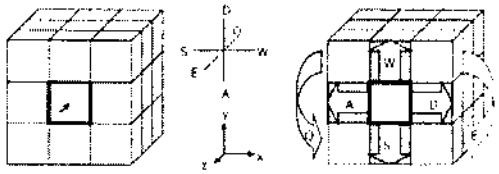


Figura 6 - Rotação dos discos usando teclado e rato

Procedimento:

1. Com o rato, selecciona-se a face referência;
2. Usando o teclado selecciona-se a direcção de deslocamento, pressionando a tecla correspondente.

Este método combina os dois anteriores, e procura minimizar as desvantagens de ambos:

- O número de teclas a usar reduz-se, permitindo uma melhor adaptação do utilizador às teclas;
- A utilização do rato reduz-se para metade pois, em vez de efectuar duas selecções basta fazer uma só.

De modo a minimizar o problema da associação entre as teclas e os discos a rodar, foi incluído um sistema de eixos (figura 6) que tem como propósito ajudar o utilizador: estes eixos representam os eixos de rotação do cubo e, como está na figura, o eixo vertical representa a rotação em torno de YY, com as teclas A e D correspondentes aos sentidos

horário e anti-horário respectivamente. O mesmo se aplica para os outros eixos.

Este foi o método adoptado, pois é o único que indica sempre a informação correcta, qualquer que seja a orientação do modelo, oferecendo melhor jogabilidade quando comparado com os métodos anteriores. Contudo, as vantagens deste método só poderão ser percebidas após alguma adaptação.

VI. *RENDERING* DO MODELO

O *rendering* ("desenho") do modelo, é feito a partir da informação de uma só face. Assim, para a construção de todo o modelo, apenas é necessário efectuar transformações geométricas a cada face, de modo a colocá-las na localização certa. Deste modo, apenas são precisos 4 vértices para desenhar a face, permitindo construir qualquer modelo independentemente da sua dimensão (em cubos).

De modo a maximizar o desempenho computacional, apenas são desenhadas as faces que estão voltadas para o exterior do modelo. Por exemplo, para um modelo de $3 \times 3 \times 3$ são desenhadas apenas 54 faces, em contraposição às 162 faces que o modelo possui na sua totalidade. Esta técnica torna-se indispensável para os imponentes modelos de $10 \times 10 \times 10$.

A. *Desenho do modelo*

As coordenadas dos vértices da face base devem corresponder à face de referência 0 (figura 2) e devem estar colocadas na parte frontal de um cubo de aresta 2 e centrado na origem (figura 7).

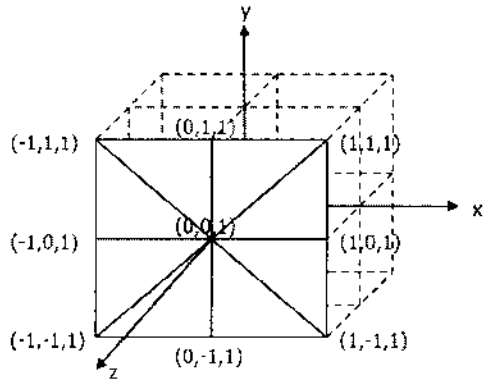


Figura 7 - Face referência para o desenho dos cubos

Serão desenhados oito triângulos por face, pelo que são necessários 9 vértices. Este número deve-se ao modelo de iluminação utilizado – "Modelo de Phong" – com sombreamento por interpolação, que favorece a qualidade do modelo com o aumento do número de subdivisões da face. A definição dos triângulos é feita por referência a um índice de uma lista de vértices (que devem ser definidos no sentido anti-horário).

Construção do modelo:

1. Rotação em torno de YY ou XX, para colocar a face na posição correcta do cubo a que pertence. Percorrendo todas as faces, o cubo é desenhado;
2. Transladar o cubo construído para a posição adequada do modelo usando os parâmetros de translação locais de cada cubo;
3. Rodar o cubo de acordo com o valor de rotação local (rotação de discos). Percorrendo todos os cubos, o modelo é desenhado;
4. Usando os parâmetros de rotação globais, rodar todo o modelo, segundo os três eixos, para os ângulos apropriados.

O resultado final, utilizando projecção perspectiva e efeitos de iluminação, é apresentado na figura 8.

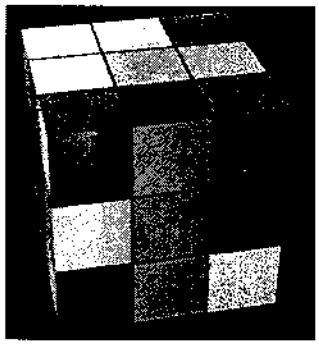


Figura 8 - O Cubo de Rubik

VII. MELHORAMENTOS FUTUROS

Pretende-se a curto/médio prazo introduzir alguns melhoramentos no que respeita à jogabilidade do Cubo de-Rubik:

- Manipulação dos discos do modelo por arrastamento do rato;
- Rotação global do modelo por arrastamento do rato;
- Incorporação de um algoritmo de auto-resolução do modelo, que permita dar uma ajuda valiosa aos iniciados nesta aventura [6].

O uso da capacidade de arrastamento do rato é uma forma de explorar as capacidades de movimento deste dispositivo, melhorando assim o nível de interacção do utilizador.

VIII. AGRADECIMENTOS

Agradece-se ao Prof. Joaquim Madeira as ideias para a realização do trabalho e o incentivo na escrita deste artigo.

REFERÊNCIAS

[1] Rubik's Cube History, http://cubeland.free.fr/infos/ernorubik.htm (Jan. 2005)

[2] Lecture Notes on the Mathematics of the Rubik's Cube, http://web.usna.navy.mil/~wdj/rubik_nts.htm (Jan. 2005)

[3] The OpenGL Oficial Site, http://www.opengl.org (Jan. 2005)

[4] E. Angel, "Interactive Computer Graphics: a top-down approach with OpenGL", 3rd ed., Addison-Wesley 2003, pp. 114–121

[5] Picking in OpenGL, http://www.soe.ucsc.edu/classes/cmps160/Fall01/picking.txt (Jan. 2005)

[6] Rubik's Cube Solution, http://www.nerdparadise.com/puzzles/333 (Jan. 2005)

Practical Issues on RF Modelling of Multi-rate Nonlinear Systems

Telmo Reis Cunha, José Carlos Pedro

Resumo **– Este artigo apresenta uma análise sobre questões práticas referentes à utilização de modelos RF de sistemas não lineares que possuem fenómenos em bandas de frequência muito distintas (denominados por sistemas multi-ritmo). Esta característica dificulta a utilização de modelos que transportam para a banda-base o comportamento do sistema, sendo aplicados os chamados modelos RF (que não consideram qualquer translação em frequência das suas características).**

Havendo um interesse cada vez maior na modelação de sistemas multi-ritmo, por exemplo, para analisar fenómenos térmicos em equipamento RF, este artigo aponta alguns problemas de ordem prática na utilização de modelos RF, nomeadamente a enorme quantidade de parâmetros necessários para o modelo, e os elevados tempos de computação associados. São também propostas algumas linhas de acção para contornar estes problemas.

Abstract **– This paper presents an analysis on practical issues regarding the use of RF modelling of multi-rate nonlinear systems. Multi-rate systems are characterized by a frequency spectrum concentrated on distinct and separated frequency bands. This characteristic makes impossible the use of base-band models (that consider frequency shifting of the system spectrum), and so, RF modelling is naturally applied.**

Noticing the growing interest on multi-rate system modelling, for example, to analyse temperature phenomena on RF equipment, this paper presents some implementation problems associated to the use of multi-rate RF models, namely the huge amount of model parameters, and the associated computation time. Some suggestions to work around these problems are also presented.

1. INTRODUCTION

The telecommunication society shows, throughout the last decades, a growing interest on nonlinear behavioural system modelling, as can be verified by the amount of published papers on the issue ([1] through [17] is just a small sample).

The predominant mathematical strategies used in nonlinear system modelling are: neural networks and multidimensional polynomial series (Volterra series). In this paper only the Volterra series approach will be considered, although the conclusions obtained from the presented analysis are extendable to the neural network case. Information on neural network modelling can be found in diverse documentation, in which [18] to [24] are some reference examples.

Volterra series are used in systems that are only mildly nonlinear, that is, whose nonlinearities can be approximated by a low order polynomial expansion around some quiescent point [8], [10], [11]. Fortunately, since nonlinearity is most of the times an undesirable source of signal fidelity impairments, many systems encountered in telecommunications and instrumentation fit that description. Equation (1) shows the Volterra series expansion, in its discrete form, of the transfer function of the system presented in Fig. 1. Notice that the Volterra series is nothing more than the sum of multidimensional convolutions between the input signal and the multidimensional impulse responses (the various n^{th}-order kernels).

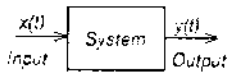


Fig. 1 - General system.

$$y(t) = \sum_{n=0}^{\infty} y_n(t) \qquad where$$
$$y_n(t) = \sum_{\tau_1=0}^{\infty} \cdots \sum_{\tau_n=0}^{\infty} h_n(\tau_1, \cdots, \tau_n)\, x(t-\tau_1) \cdots x(t-\tau_n) \tag{1}$$

Both representations – neural networks and Volterra series – have advantages and disadvantages. For example, the parameters of a neural network result from a training procedure performed over the application of a known excitation sequence to its input. Different parameter sets can result if different training sequences are used, so their predictive ability can be compromised if the model input has some characteristics that fall outside the expected behaviour they were supposed to have. Volterra series, similarly to one-dimensional polynomials, have severe convergence problems when the input exceeds a certain interval around the quiescent point.

It is not the objective of this paper to analyse theoretical limitations of such methods, being the reader suggested to [8], [10], [11] for further details on the subject. This article is more concerned with practical implementation aspects of nonlinear FIR filters when multi-rate behaviour is to be modelled.

Taking as an example a power amplifier working at some RF frequency band, its behaviour is likely to change if the temperature of its components varies. In many cases, such behavioural changes are quite noticeable, being worthy to model. Since temperature variation is clearly a very low

frequency phenomena, when compared to the RF working band of the amplifier, this becomes a multi-rate system. Evidently, there are many other equipment and phenomena that present this multi-rate characteristic.

If multi-rate information is to be considered by the model, it makes no sense to use a low-pass equivalent model, otherwise part of the information would be lost. So, the solution adopted by several authors is to model the system without any frequency shifting scheme, resulting in the so called RF model.

Since the objective of these models is to simulate the system behaviour to a broad class of inputs, the parameters required for its characterization are the time-domain parameters: the impulse response (first-order kernel) in the case of a linear system, or the multidimensional kernels of the Volterra series (1) in the nonlinear case.

This paper shows that the number of parameters of the general time-domain Volterra series of a multi-rate system is huge, which can compromise the practical use of such technique if some restricting strategies are not applied. A large set of parameters means, on one hand, that the system identification process is very hard and, on the other hand, the computation time during simulation can rapidly become unbearable.

The following Section demonstrates that, for a multi-rate linear system (and also for the nonlinear case), the respective impulse response has a higher set of parameters than the frequency-domain representation, although both contain the same information. Some system examples are given to illustrate the inefficiency of these models.

The third Section of this paper suggests a strategy to avoid having such a huge number of parameters to deal with. This is achieved by restricting the space of non-zero parameters, using a predefined model topology. An example with first and third-order kernels will be thoroughly analysed.

II. TIME DOMAIN REPRESENTATION OF MULTI-RATE SYSTEMS

It is known that the time domain representation of a linear system is the system impulse response, which is the inverse Fourier transform of the frequency-domain representation. In the present case, the Discrete Fourier Transform (DFT) will be considered.

Let us analyse the linear system whose frequency-domain representation is given in Fig. 2 (only the amplitude is depicted). Applying the inverse DFT results in the system impulse response shown in Fig. 3.

From the Fourier transform properties, the following remarks are taken:

- the sampling period Δt of $h(t)$ is the inverse of twice the maximum frequency of $H(j\omega)$;
- the period T of $h(t)$ is the inverse of Δf, which is equivalent to saying that T can be small if $H(j\omega)$ is smooth.

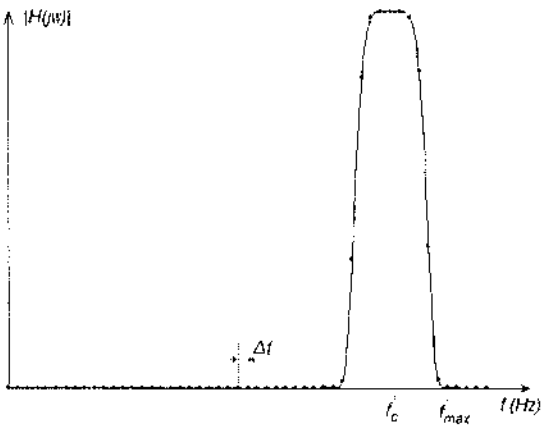


Fig. 2 - Frequency domain representation of a linear system.

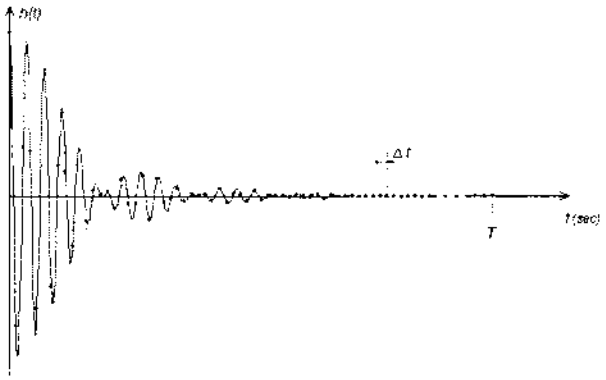


Fig. 3 - Time domain representation of the system of fig.2.

What is immediately observed in Fig. 2 and Fig. 3 is that the frequency-domain representation requires only the knowledge of 11 non-zero parameters while the time domain representation requires at least 36! But both should represent the same information, so why is the number of non-zero parameters so different?

Let us look at the information that each representation contains. As is known, the envelope of $h(t)$ is nothing but the inverse DFT of $H(j\omega)$ when it is shifted to base-band (through a band-pass sampling process [25], [26], for example). Then, $h(t)$ can be formed by filling the inside of the envelope with a cosine wave at the centre frequency f_c of $H(j\omega)$ (obviously, the cosine wave is multiplied by the envelope), and sampled with period Δt.

So, the information of each representation can be described by:

Frequency Domain:

- Curve parameterization of $H(j\omega)$ (by means of a polynomial, for example) → n parameters;
- Centre frequency f_c → 1 parameter;
- Bandwidth Bw → 1 parameter;
- Frequency spacing Δf → 1 parameter;
- All values outside $[f_c - B/2, f_c + B/2]$ are zero (this statement is information too);
- $H(j\omega)$ has even symmetry on the amplitude and odd symmetry on the phase.

Time Domain:

- Curve parameterization of the envelope of $h(t)$ (by means of a polynomial, for example) → n parameters;
- Frequency f_c of the cosine wave multiplying the envelope → 1 parameter;
- Sampling period Δt → 1 parameter;
- Period T of $h(t)$ → 1 parameter;
- $h(t)$ is real and has even symmetry.

The similarities are evident, and the only difference is that $h(t)$ has no statement saying that a certain number of its values are zero! This means that the samples $h(t_k)$ can be non-zero, for all t_k in the period T!

This result, as can be verified through equation (2), is an evident consequence of the definition of the inverse DFT.

$$h(t_k) = \sum_{i=-\infty}^{\infty} H(\omega_i)\, e^{j\omega_i t_k} \qquad (2)$$

On systems like the one presented in Fig. 2, or even in systems having fundamental and harmonic bands, some schemes are usually implemented to avoid handling of a huge number of time domain parameters. As shown in Fig. 4, a sub-sampling (a procedure called band-pass sampling) followed by a proper low pass filtering can concentrate all the non-zero information of $H(j\omega)$ at low frequencies, which naturally produces an impulse response with fewer parameters than the original $H(j\omega)$ (Δt is wider and T remains the same).

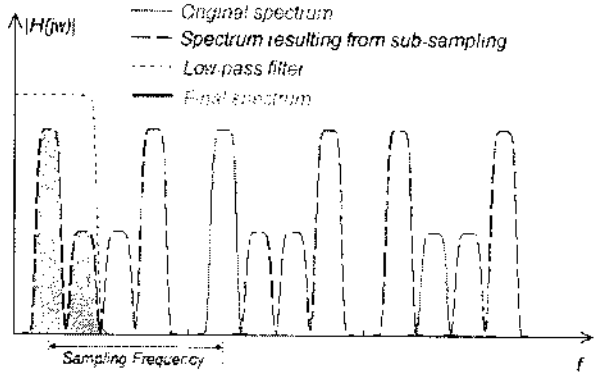


Fig. 4 - Band-pass sampling example, followed by low-pass filtering.

If the system has non-negligible multi-rate characteristics where the location of the distinct bands does not follow a general relationship (such as with the harmonic bands), it can be complicated to apply a frequency down-shifting scheme. Again, if multi-rate characteristics are to be preserved by the model, the low-pass equivalent model is not a valid option. A solution adopted by several authors is simply to use the RF model, where no frequency shift is considered. A typical situation is in the analysis of temperature effects (very low frequency) on systems that operate at much higher frequencies.

Take, for example, an equipment working at VHF, say with f_c=100MHz and a bandwidth of 100kHz, presenting a smooth curve that allows its parameterization with a frequency spacing of Δf=5kHz. The inverse DFT of this band has Δt≈5ns and T=200μs, which results in a top limit of 40,000 parameters to be used in the time-domain simulation. This number is already huge, but if some low frequency phenomena is also to be considered in the model, say at base-band with cut-off frequency at 2kHz, and with a pattern that requires a frequency spacing of 20Hz to be properly represented, then the total system impulse response would have T=50ms, keeping the previous sampling period Δt=5ns. The number of time domain parameters is then 10,000,000 (250 times more)! Fig. 5 illustrates this through an example of a system that clearly has slow and fast time scale characteristics.

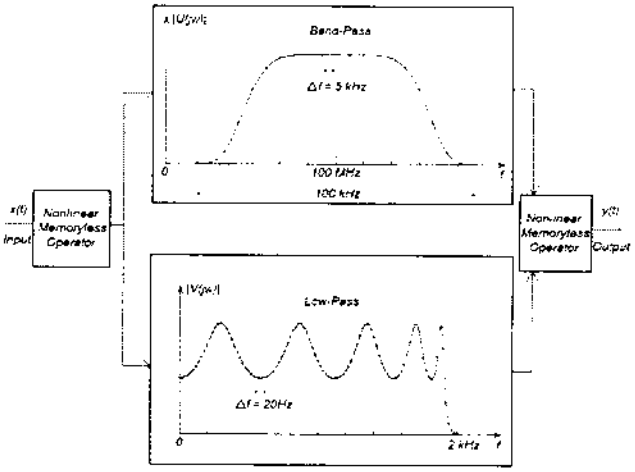


Fig. 5 - Example of a multi-rate system.

For a general multi-rate RF linear system, the number of time domain parameters is given by (3). It is equivalent to represent the full frequency span (from $-f_{máx}$ to $f_{máx}$) with the minimum frequency spacing $\Delta f_{mín}$.

$$Number\ of\ Parameters = \frac{T_{max}}{\Delta t_{min}} = \frac{2\, f_{max}}{\Delta f_{min}} \qquad (3)$$

Let us now consider the nonlinear case, restricting the nonlinearities up to the third-order kernel. The system model is given by equation (4).

$$\begin{aligned} y(t) = & \sum_{\tau=0}^{N-1} h_1(\tau) x(t-\tau) + \\ & + \sum_{\tau_1=0}^{M-1} \sum_{\tau_2=0}^{Q-1} h_2(\tau_1,\tau_2)\, x(t-\tau_1)\, x(t-\tau_2) + \\ & + \sum_{\tau_1=0}^{P-1} \sum_{\tau_2=0}^{Q-1} \sum_{\tau_3=0}^{R-1} h_3(\tau_1,\tau_2,\tau_3)\, x(t-\tau_1)\, x(t-\tau_2)\, x(t-\tau_3) \end{aligned} \qquad (4)$$

It is evident that $h_1(\tau)$ is the impulse response of the linear approximation of the system, to which corresponds $H_1(j\omega)$ in the frequency domain, after applying the DFT.

The second-order kernel $h_2(\tau_1,\tau_2)$ has a 2D domain, and $h_3(\tau_1,\tau_2,\tau_3)$ a 3D domain. They also have the respective frequency-domain representations $H_2(j\omega_1,j\omega_2)$ and $H_3(j\omega_1,j\omega_2,j\omega_3)$ which result from applying the multidimensional DFT [27].

Looking into the properties of the multidimensional DFT, it is observed that it preserves the same characteristics of the DFT, except that it is extended to a higher dimension domain. So, the problem of having a huge number of time domain parameters remains, and it gets even worse due to the domain dimension – that is, if $h_2(.)$ has a memory span of N elements on both τ_1 and τ_2 axes, it contains N^2 parameters, and if M is the memory span of $h_3(.)$ then it has M^3 time domain parameters!

If (4) is used as a RF model of a multi-rate nonlinear system, then it is necessary to constrain de domain of each kernel so that it gets highly reduced. Otherwise, there will be such a huge number of non-zero parameters that:

- the parameter identification process will probably be an impossible task;
- the amount of computer memory required to store the model is huge;
- the computation time during simulation is tremendous, even in high performance computers.

III. RESTRICTING THE NUMBER OF MODEL PARAMETERS

Fig. 5 already shows an example of a model constraint: the frequency domain representation has two clusters of non-zero parameters; all others are null. But, to what correspond these clusters in the respective time domain representation?

Fig. 6 shows the inverse DFT of both base-band and RF clusters. Since the responses of both low-pass and band-pass filters are supposed to interact, $v(t)$ is sampled at the same sampling frequency as $u(t)$, so $v(t)$ is represented with redundancy (we are considering that the model topology of Fig. 5 is not known a priori). This means that $v(t)$ can be approximated by a step function as shown in Fig. 7. In other words, $v(t)$ can be represented by 200 distinct parameters where each one is then repeated 50,000 consecutive times. This reduces, undoubtedly, the parameter identification burden, but in terms of simulation it still requires the convolution of all the 10 Msamples with the filters input signal.

This simulation process can also be simplified by grouping the input signal in 200 moving sums, each of which is multiplied by the respective parameter of $v(t)$ to perform the convolution. At each time instant, each moving sum only needs to add a new sample and subtract the tail sample, to the moving sum result of the previous epoch.

But it is also necessary to extract and use in simulation the 40,000 parameters of $u(t)$! If $U(j\omega)$ can have a smoother pattern then Δf can be higher, reducing the number of parameters in $u(t)$.

This constraint procedure, with the creation of clusters in the frequency-domain representation, can be extended to the nonlinear case. It is simply required to create n-dimensional clusters in the $H_n(j\omega_1,\ldots,j\omega_n)$ frequency domain kernels. But, contrary to the linear case, this is not an intuitive procedure in practice since multi-frequency signals are now being considered.

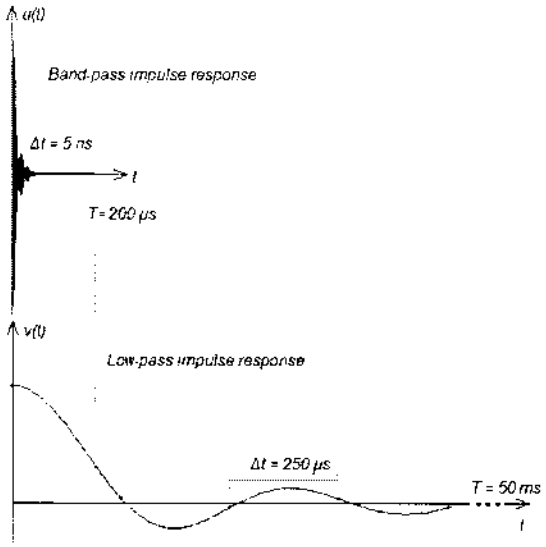


Fig. 6 - Impulse response of each band-pass and low-pass clusters.

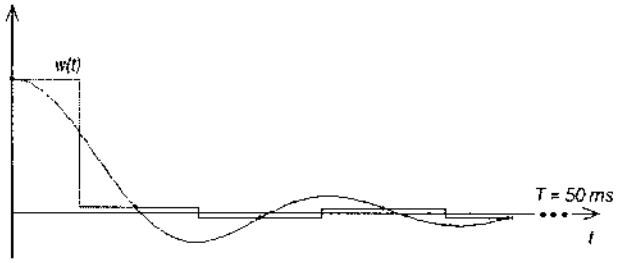


Fig. 7 - Stepwise approximation of v(t).

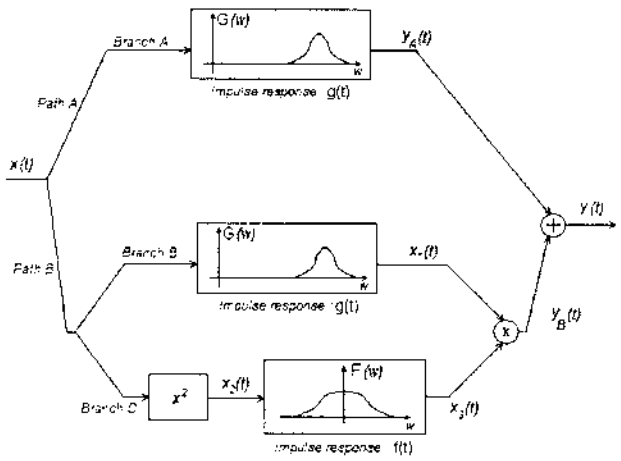


Fig. 8 - Example of a model topology.

Another way of restricting the information contained in a system is to use a model topology. Saying that a certain system fits the description of some model topology is, in

fact, an imposition on the information of the system that will be represented by that model. An example of what is called behavioural modelling with a priori knowledge of the system.

Let us consider an example of a model topology for a nonlinear system, given by Fig. 8.

This topology presents two main paths – path A models the linear behaviour of the system, which corresponds to the first-order kernel of (1); path B models the nonlinear characteristic of the system, which, in this case, is concentrated on the third-order kernel (its output is a double product of the input).

An immediate observation is that this topology restricts its usefulness to systems whose kernel orders other than first and third only have non-significant information. Nevertheless, it is known that several nonlinear interesting phenomena, in telecommunication systems and other areas, have most of their relevant information in those two kernels.

Notice that this topology can model multi-rate systems, as will be shown later, where low frequency components can influence the behaviour at in-band frequencies.

This model is then defined by:

- the model topology of Fig. 8;
- the N parameters of the band-pass linear filter of branches A and B – $g(t)$;
- the M parameters of the low-pass linear filter of branch B – $f(t)$.

So, given this topology, it should only be required $N+M$ parameters to model a system (and to simulate it).

Let us analyse the time-domain response of such topology, given a general input signal $x(t)$. Equation (5) shows the output of path A, and equation (6) the output of path B.

$$y_A(t) = \sum_{i=0}^{N-1} g(i)\, x(t-i) \tag{5}$$

$$y_B(t) = \sum_{i=0}^{N-1}\sum_{j=0}^{M-1} g(i)\, f(j)\, x(t-i)\, x^2(t-j) \tag{6}$$

By matching the terms of equation (1) with equation (6) it becomes clear that (6) is the third-order kernel of a Volterra series expansion. Moreover, (7) gives a general definition of the third-order kernel that fits (6), where a restriction on the domain of the kernel is evident, as depicted in Fig. 9.

$$h_3(i,j,k) = \begin{cases} g(0)\cdot f(0) & , \ se \ \ i=j=k \\ \frac{1}{3}g(i)\cdot f(j) & , \ se \ \ i\neq j=k \\ \frac{1}{3}g(j)\cdot f(k) & , \ se \ \ j\neq k=i \\ \frac{1}{3}g(k)\cdot f(i) & , \ se \ \ k\neq i=j \\ 0 & , \ se \ \ i\neq j\neq k \end{cases} \tag{7}$$

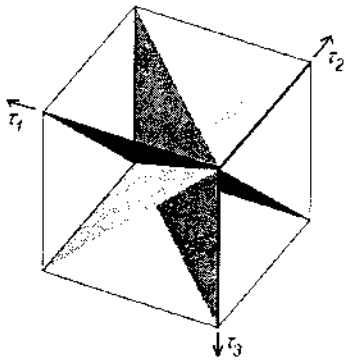


Fig. 9 - Domain of the h_3 time-domain kernel.

Notice that, for symmetry purposes, the term $g(i)f(j)$ was distributed evenly by the three admissible planes. Other distribution would also verify the matching with (6) as long as the sum of the three contributions would equal 1.

Assuming no particular specification to $g(t)$ and $f(t)$, it is obvious from (7) that the $h_3(\tau_1,\tau_2,\tau_3)$ kernel is formed by a bi-dimensional matrix that is copied into the three planes represented in Fig. 9. Taking, for example, $g(t)=u(t)$ of Fig. 6, and $f(t)=w(t)$ depicted in Fig. 7, this bi-dimensional matrix A, whose element (i,j) is given by $g(i)f(j)$, is plotted in Fig. 10. It is shown that each row equals the function $g(t)$ multiplied by a constant that changes only every 50,000 columns.

Similarly to what was exposed regarding the system of Fig. 5, the clustered information ($g(t)$ and $f(t)$ occupy two distinct and very separated frequency bands) is visible in the pattern of the matrix of Fig. 10. Again, it requires only $N+M$ parameters to be determined instead of $N\times M$ parameters (or even instead of the full cube of the $h_3(.)$ domain).

Continuing with the analysis in the time-domain, it is easy to observe that with the *multiple impulse input* identification method [8], the $N+M$ parameters of a system can be easily extracted without having to sweep the entire A matrix. In more general terms, if a certain model topology can be applied, then a specific and dedicated identification process can be used to extract the model parameters.

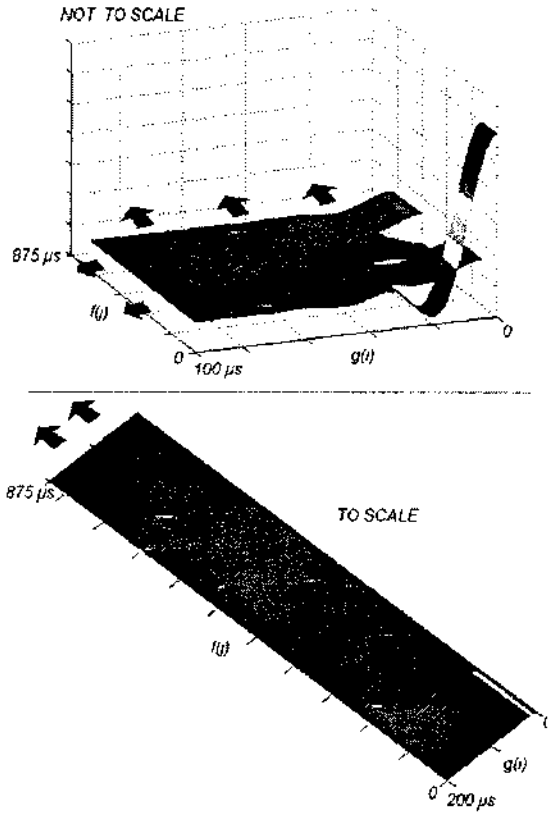


Fig. 10 - Bi-dimensional matrix A (only the upper envelope of $u(t)$ was considered).

In terms of simulation, two strategies are suggested to avoid making a large number of multiplications per epoch. The first is to implement an algorithmic scheme similar to the moving sum presented above. The second strategy, which looks more attractive, is to take advantage of the model topology. In fact, since the parameters of $g(t)$ and $f(t)$ can be identified separately, each branch of Fig. 8 can be evaluated independently, being the results of branch B and C multiplied and added to the result of branch A (moreover, the result of branch A is equal to the one of branch B).

Analysing, now, the topology of Fig. 8 in the frequency domain, it becomes again evident the restrictions that this topology imposes on the $H_3(.)$ kernel.

Using the *harmonic input method* [8], consider a three-tone input to path B, as shown in (8).

$$x(t) = e^{j\omega_1 t} + e^{j\omega_2 t} + e^{j\omega_3 t} \tag{8}$$

After proceeding with the math analysis of path B, and noticing that the final product is processed as a convolution in the frequency domain, expression (9) is reached. Then, the $H_3(j\omega_1, j\omega_2, j\omega_3)$ frequency-domain kernel is given by (10), according to the *harmonic input method.*

If $g(t)$ is a band-pass filter and $f(t)$ a low-pass filter, like those defined in Fig. 5, then $H_3(j\omega_1, j\omega_2, j\omega_3)$ has non-zero values on the coloured volumes depicted in Fig. 11. This evidently shows the restrictions that this topology imposes on the domain of $H_3(.)$.

$$\begin{aligned}
Y(\omega) = {} & G(\omega_1)\left(\begin{matrix} F(2\omega_1)e^{j(2\omega_1)} + F(2\omega_2)e^{j(\omega_1+2\omega_2)} + \\ + F(2\omega_3)e^{j(\omega_1+2\omega_3)} \end{matrix}\right) + \\
& + G(\omega_2)\left(\begin{matrix} F(2\omega_1)e^{j(2\omega_1+\omega_2)} + F(2\omega_2)e^{j(3\omega_2)} + \\ + F(2\omega_3)e^{j(\omega_2+2\omega_3)} \end{matrix}\right) + \\
& + G(\omega_3)\left(\begin{matrix} F(2\omega_1)e^{j(2\omega_1+\omega_3)} + F(2\omega_2)e^{j(2\omega_2+\omega_3)} + \\ + F(2\omega_3)e^{j(3\omega_3)} \end{matrix}\right) + \\
& + 2G(\omega_1)\left(\begin{matrix} F(\omega_1+\omega_2)e^{j(2\omega_1+\omega_2)} + F(\omega_1+\omega_3)e^{j(2\omega_1+\omega_3)} + \\ + F(\omega_2+\omega_3)e^{j(\omega_1+\omega_2+\omega_3)} \end{matrix}\right) + \\
& + 2G(\omega_2)\left(\begin{matrix} F(\omega_1+\omega_2)e^{j(\omega_1+2\omega_2)} + F(\omega_1+\omega_3)e^{j(\omega_1+\omega_2+\omega_3)} + \\ + F(\omega_2+\omega_3)e^{j(2\omega_2+\omega_3)} \end{matrix}\right) + \\
& + 2G(\omega_3)\left(\begin{matrix} F(\omega_1+\omega_2)e^{j(\omega_1+\omega_2+\omega_3)} + F(\omega_1+\omega_3)e^{j(\omega_1+2\omega_3)} + \\ + F(\omega_2+\omega_3)e^{j(\omega_2+2\omega_3)} \end{matrix}\right)
\end{aligned} \tag{9}$$

$$\begin{aligned}
& H_3(\omega_1,\omega_2,\omega_3) = \\
& = \frac{1}{3}\left(G(\omega_1)F(\omega_2+\omega_3) + G(\omega_2)F(\omega_1+\omega_3) + G(\omega_3)F(\omega_1+\omega_2)\right)
\end{aligned} \tag{10}$$

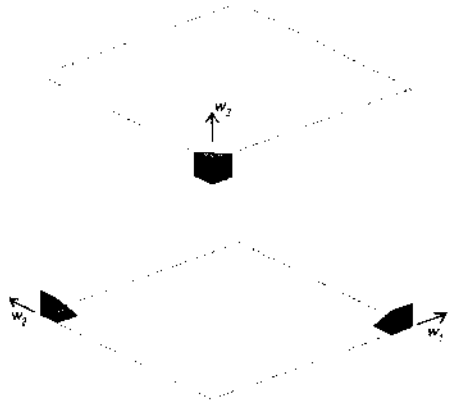


Fig. 11 - Domain of the H3 frequency-domain kernel.

VI. CONCLUSIONS

When multi-rate phenomena are being studied in a nonlinear (or even on a linear) system, some care must be taken when an RF model of the system is considered and implemented by means of a nonlinear FIR filter. As shown in this paper, the number of model parameters easily grows to incredible values that obviate any system identification procedure, and deeply compromises the computation time required to simulate the system behaviour.

To prevent that, some strategy must be used to strongly restrict the domain of the nonlinear FIR filter kernels. A suggested approach is to consider a specific a priori topology for the model. The model topology is, by itself, a

specification of how the system relevant information is (or should be) distributed. A proper model topology can largely reduce de number of parameters necessary to model the system, and also to significantly reduce the epoch-to-epoch simulation time. Dedicated parameter extraction procedures can also be specified to ease the identification process required to determine the parameter values that best model a certain system (according to that topology).

However, to guarantee the desired model predictive capabilities, when using a determined topology it is necessary to verify that it can, in fact, represent the system behaviour.

REFERENCES

[1] V. Rizzoli and A. Neri, "State of the Art and Present Trends in Nonlinear Microwave CAD Techniques", IEEE Trans. on Microwave Theory and Tech., Vol. MTT-36, No. 2, pp.343-365, 1988.

[2] M. S. Nakhla and J. Vlach, "A Piecewise Harmonic Balance Technique for Determination of Periodic Response of Nonlinear Systems", IEEE Trans. on Circuits and Systems, Vol. CAS-23, No. 2, pp.85-91, 1976.

[3] V. Rizzoli, A. Neri and F. Mastri, "A Modulation-Oriented Piecewise Harmonic Balance Technique Suitable for Transient Analysis and Digitally Modulated Analysis", 26th European Microwave Conference Proc., pp.546-550, Prague, 1996.

[4] J. C. Pedro and N. B. Carvalho, "Simulation of RF Circuits Driven by Modulated Signals Without Bandwidth Constraints", 2002 IEEE International Microwave Symposium Dig., Seattle, 2002.

[5] D. Sharrit, "New Method of Analysis of Communication Systems", IEEE MTT-S Nonlinear CAD Workshop, 1996.

[6] L. Chua, "Nonlinear Circuits", IEEE Trans. on Circuits and Systems, Vol. CAS-31, No. 1, pp.69-87, 1984.

[7] V. Mathews and G. Sicuranza, *Polynomial Signal Processing*, John Wiley & Sons, Inc., New York, 2000.

[8] J. C. Pedro and N. B. Carvalho, Intermodulation Distortion in Microwave and Wireless Circuits, Artech House, Norwood, 2003.

[9] T. J. Aprille and T. N. Trick, "Steady-State Analysis of Nonlinear Circuits With Periodic Inputs", Proceedings of the IEEE, Vol. 60, No.1, pp.108-114, 1972.

[10] S. A. Maas, Nonlinear Microwave Circuits, Artech House, Norwood, MA, 1988.

[11] M. Schetzen, The Volterra and Wiener Theories of Nonlinear Systems, John Wiley & Sons, New York, 1980.

[12] N. B. Carvalho and J. C. Pedro, "Multi-tone Frequency Domain Simulation of Nonlinear Circuits in Large and Small Signal Regimes", IEEE Trans. on Microwave Theory and Tech., Vol. MTT-46, No. 12, pp.2016-2024, 1998.

[13] C. R. Chang and M. B. Steer, "Frequency-Domain Nonlinear Microwave Circuit simulation Using the Arithmetic Operator Method", IEEE Trans. on Microwave Theory and Tech., Vol. MTT-38, No. 8, pp.1139-1143, 1990.

[14] V. Rizzoli, F. Mastri, E. Furini and A. Costanzo, "A Krylov-Subspace Technique For The Global Stability Analysis of Large Nonlinear Microwave Circuits", 2001 IEEE International Microwave Symposium Dig., pp.435-438, 2001.

[15] N. B. Carvalho and J. C. Pedro, "Analysis and Measurement of Multi-tone Intermodulation Distortion of Microwave Frequency Converters", 2001 IEEE International Microwave Symposium Dig., pp. 1671-1674, 2001.

[16] D. Hente and R. H. Jansen, "Frequency Domain Continuation Method for the Analysis and Stability Investigation of Nonlinear Microwave Circuits", IEE Proceedings-H Microwaves Antennas and Propagation, Vol. 133, No. 5, pp. 351-362, 1986.

[17] P. J. Rodrigues, Computer Aided Analysis of Nonlinear Microwave Circuits, Artech House, Inc., Norwood, MA, 1998.

[18] Q. J. Zhang and K. C. Gupta, *Neural Networks for RF and Microwave Design*, Artech House, Norwood, 2000.

[19] G. Cybenko, "Approximation by Superpositions of a Sigmoidal Function", *Math. Control Signals Systems*, vol. 2, pp.303-314, 1989.

[20] K. Hornik, M. Stinchcombe and H. White, "Multilayer Feedforward Networks are Universal Approximators", *Neural Networks*, vol. 2, pp.359-366, 1989.

[21] Y. Fang, M. C. Yagoub, F. Wang and Q. J. Zhang, "A New Macromodeling Approach for Nonlinear Microwave Circuits Based on Recurrent Neural Networks", *IEEE Trans. on Microwave Theory and Tech.*, vol. MTT-48, pp.2335-2344, Dec. 2000.

[22] D. Schreurs, N. Tufillaro, J. Wood, D. Usikov, L. Barford and D. E. Root, "Development of Time Domain Behavioural Non-Linear Models for Microwave Devices and ICs from Vectorial Large-Signal Measurements and Simulations", *Europ. Gallium Arsenide and other Semiconductors Applications Symp. Dig.*, pp.236-239, Oct. 2000.

[23] V. Rizzoli, A. Neri, D. Masotti and A. Lipparini, "A New Family of Neural Network-Based Bidirectional and Dispersive Behavioral Models for Nonlinear RF/Microwave Subsystems", *Int. Jour. of RF and Microwave CAE*, vol. 12, pp.51-70, 2002.

[24] J. Xu, M. Yagoub, R. Ding and Q. J. Zhang, "Neural-Based Dynamic Modeling of Nonlinear Microwave Circuits", *IEEE Trans. on Microwave Theory and Tech.*, vol. MTT-50, pp.2769-2780, Dec. 2002.

[25] F. H. Harris, Multirate Signal Processing for Communication Systems, Prentice Hall PTR, 2004.

[26] A. Oppenheim and R. Schafer, Discrete-Time Signal Processing, Prentice Hall, Englewood Cliffs, 1999.

[27] R. Tolimieri, M. An and C. Lu, Mathematics of Multidimensional Fourier Transform Algorithms, Springer Verlag, New York, 1997.

Development and operation of a Bluetooth demonstrator

Pedro Duarte, José Alberto Fonseca, Paulo Bartolomeu

Abstract – **Wireless communications in distributed and/or embedded systems is an important research topic today. One of the emerging standards in this domain is Bluetooth. In this paper a demonstrator of embedded devices connected with Bluetooth is presented. The demonstrator is used to show the development technology available at our laboratory. So, besides the description of its operation, the paper includes an overview of the Bluetooth standard and a step by step presentation of the implementation of the modules using one of the current development tools: BlueCore from the company Cambridge Silicon Radio.**

I. INTRODUCTION

Bluetooth is a protocol for wireless communication among small embedded devices which has become popular in the last few years. The research group at the Laboratory of Electronic Systems of IEETA has been working with Bluetooth modules since two years, starting with applications concerning environment monitoring and, currently, working in the wireless transmission of music in MIDI format. In order to illustrate the potential of development it was decided to build a demonstrator that can be used to show the operation of Bluetooth to interconnect two embedded modules or to interconnect embedded modules to a personal computer.

In this paper this demonstrator is described, either in what concerns its development or in what concerns its operation. This paper can then also be used as a user manual of the demonstrator.

The paper is organized in 6 sections. In the next section an overview of Bluetooth is given. The third section describes the architecture of the demonstrator. Section IV includes the description of the operation in a form suitable for a user. Section V gives some implementation details and section VI concludes the paper.

II. AN OVERVIEW OF BLUETOOTH

Bluetooth is an open standard for radio-based communications in the 2.4 GHz Industrial Scientific and Medical (ISM) band targeting low power, low cost, low range and moderate rate applications.

The Bluetooth technology appeared in 1999 as a Bluetooth SIG specification [1][2]. In 2001 Bluetooth specification 1.1 was released [3]. Later, in 2002, the IEEE 802.15 Group adopted this specification (with minor changes) as an IEEE standard, the IEEE 802.15.1 [4]. In 2003, the Bluetooth 1.2 version was officially released [5] and recently, the Bluetooth SIG adopted the new 2.0 Bluetooth specification [6]. Three major companies are supporting the Bluetooth 2.0 specification, namely Cambridge Silicon Radio (CSR) [7], Broadcom [8] and RF Micro Devices (RFMD) [9].

The Bluetooth specification defines the protocol stack shown in Figure 1. This protocol stack can be divided in three logical groups [10]: the application protocol group, the middleware protocol group and the transport protocol group.

The application protocol group consists on the applications (Bluetooth-aware or not) that use the Bluetooth technology.

The middleware protocol group consists on both Bluetooth specific protocols like the serial port emulation (RFCOMM) and other adopted protocols like the Object Exchange Protocol (OBEX).

Finally, the transport protocol group consists of protocols exclusively developed for the Bluetooth technology like Logical Link Controller and Adaptation Protocol (L2CAP) or the Host Controller Interface (HCI).

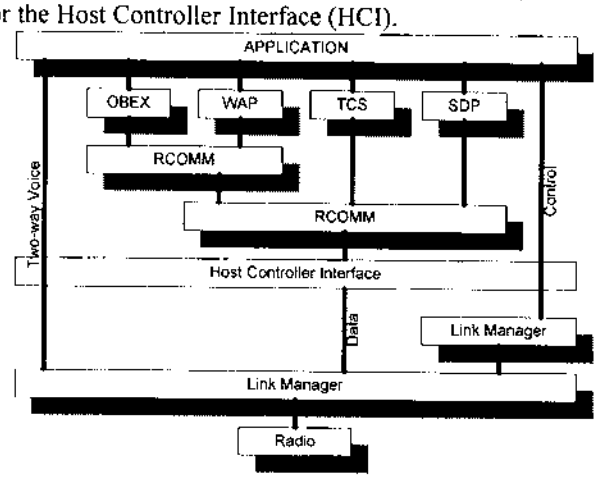


Figure 1 – Bluetooth Stack (Source: Robert Morrow)

The Logical Link and Adaptation Protocol (L2CAP) provides a transparent packet interface to higher layers through protocol multiplexing and packet segmentation (and reassembly). This protocol is also responsible for maintaining the negotiated Quality-of-Service (QoS) when applicable.

The Host Controller Interface (HCI) protocol isn't (as the name suggests) a protocol, but rather an interface allowing

a host device to access the lower Bluetooth stack protocols.

The Link Manager Protocol (LMP) is established between two Link Manager entities, to manage the air-interface link properties between them. This protocol manages security, power and bandwidth usage.

The Bluetooth Baseband layer specifies all the procedures required to establish a Bluetooth link between two devices. It also specifies the network topology and defines the bandwidth sharing mechanism between devices in a piconet which is defined as the network consisting of a master device and at least one slave. One piconet can accommodate up to seven active slaves.

Baseband defines a Time Division Duplex (TDD) scheme to enable device communication. In this sense the piconet master transmits in specific slots and a slave can only transmit if the master has polled it.

Baseband defines two types of Bluetooth links between devices: Asynchronous Connectionless (ACL) and Synchronous Connection-Oriented (SCO). A piconet supports only one ACL link between a master and a slave while a maximum of three SCO links can be established between them.

ACL links carry best-effort traffic and are suited for asynchronous transmissions. Data integrity is assured by retransmission, sequence and forward error correction (FEC) mechanisms.

SCO links support periodic data transmissions at a 64Kb/s rate in each direction. SCO traffic can't be retransmitted and thus can only recover from errors by using FEC mechanisms. The latest Bluetooth specifications 1.2 and 2.0 define a new type named extended SCO (eSCO) that allows for retransmissions.

The radio layer defines the Bluetooth's radio transceiver characteristics, which operates in the 2.4 GHz licence-free ISM band using a Frequency Hopping Spread Spectrum (FHSS) technique through 79 one-MHz channels. This is performed using a pseudo-random sequence derived from the piconet master's address at a rate of 1600 hops/sec. The specifications 1.2 and 2.0 provide an additional feature named Adaptative Frequency Hopping (AFH) which improves Bluetooth co-existence with other wireless technologies (e.g. Wi-Fi).

Three classes of radio devices are defined: class 1 radios that can transmit up to 100 mw (≈100m range); class 2 radios up to 2.5 mw (≈10m range) and class 3 radios up to 1 mw (≈10cm range).

Additional information concerning the connection establishment can be found in [11]. A deeper overview of the Bluetooth standard can be obtained in [12].

III. ARCHITECTURE OF THE DEMONSTRATOR

A. Bluetooth embedded nodes

The most important parts of the demonstrator are the embedded nodes. Each node is based on a Bluetooth module fabricated by the company Airlogic [13] which we call ABM (from Airlogic Bluetooth Module). This module integrates a core made by the company CSR (Cambridge Silicon Radio) [12] which is called BlueCore2 and includes a specific 16 bits RISC processor called XAP2, a flash memory and all the hardware interfaces required to implement the lower levels of the Bluetooth hardware (figure 2). The ABM module offers also a collection of different digital interfaces that can be used either for configuration or for operation purposes. The module includes also two analog to digital conversion inputs (figure 3).

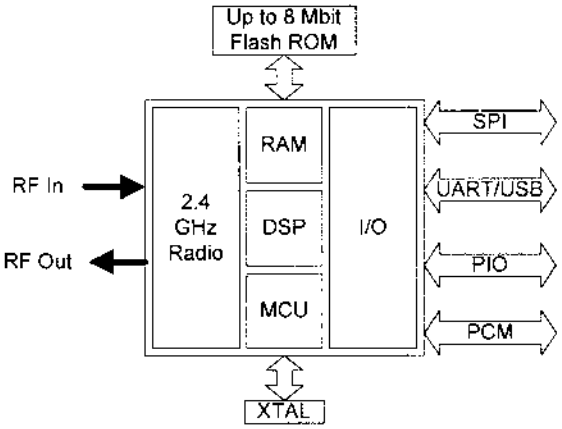


Figure 2 – Internal architecture of the CSR BlueCore2 core

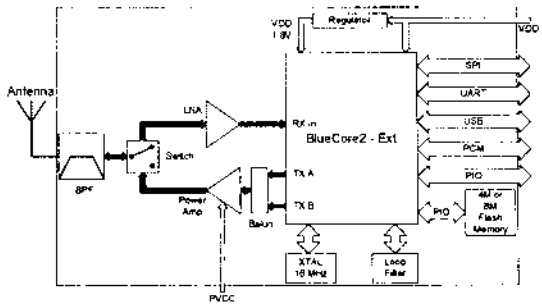


Figure 3 – The ABM - Airlogic Bluetooth Module

The CSR company offers to Bluetooth developers a software tool called BlueSuite that can be used to program the firmware or user application and to configure the ABM (or other CSR) modules. Another tool that is offered is BlueLab that is aimed to the development of integrated applications and/or higher level of Bluetooh stack integration, e.g., a RFCOMM stack interface, a complete SPP profile or a user application on the module. As an example, one could develop a program that reads from one of the A/D inputs and writes the value periodically to an emulated serial port using a small user program and the SPP profile.

The ABM module constitutes the base of our B2EN module (Basic Bluetooth Embedded Node). B2EN includes a set of buttons to interface with the user, a RS232 level adapter, power conditioning circuitry, connectors and a 7-segment display (figure 4).

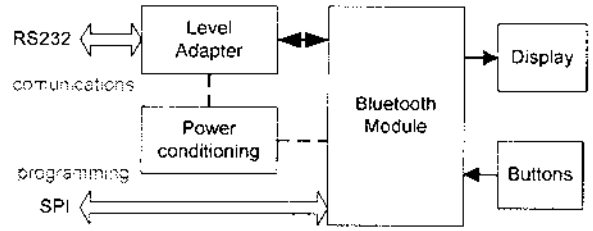


Figure 4 – Block diagram of the B2EN node.

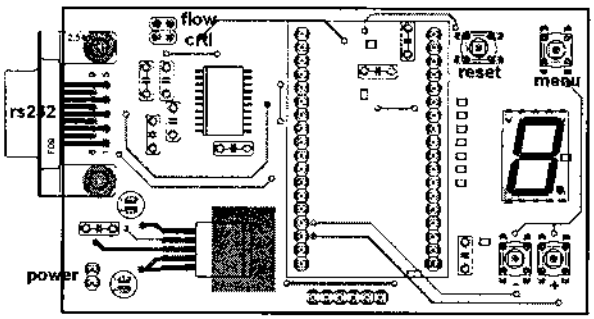


Figure 5 – Layout of the B2EN node.

Due to the modules used, the B2EN module is a Bluetooth Class 1 device with a 100 meters range. It can be powered between 4 and 15 Volts. Currently it is powered with 4 AA batteries.

B. Architecture of the demonstrator

The demonstrator offers two possible interconnections:

a) Interconnection of two B2EN modules which can communicate in different ways (Figure 6).

b) Interconnection of one or several B2EN modules to a personal computer equipped with a Bluetooth interface using the Serial Port profile (Figure 7).

Figure 6 – Interconnection of two B2EN modules

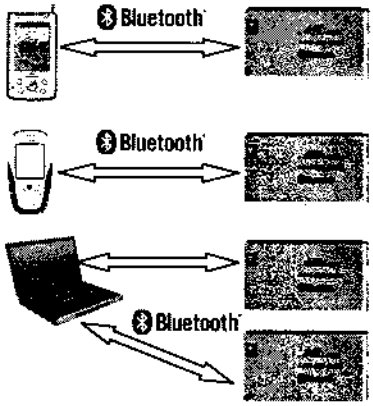


Figure 7 - Interconnection of one or several B2EN modules to a personal computer

IV. OPERATION OF THE BLUETOOTH DEMONSTRATOR

A. Modes of operation of the B2EN modules

Our Bluetooth demonstrator is designed to be used in two types of demonstration: visual demonstration and serial port demonstration. In the visual demonstration the 7-segment display is used to show the operation. In the serial port demonstration one has to connect the serial port interfaces of the B2EN modules.

Each B2EN module has then 3 possible modes of operation (their pairing will be explained latter):

Mode "1" – The B2EN module sends to the 7-segment display every character it has received through the Serial Port Profile. Just the 0-9 and A-F ASCII characters are displayed. All others are ignored. The buttons are not used here for operation.

Mode "2" – The value shown in the display can be incremented or decremented by means of the "+" and "-" buttons. Each time a change is performed, the B2EN module sends the character to the module it is connected to, using the Serial Port Profile.

Mode "3" – The information received at the RS232 port of the node is sent to the other node using the Serial Port Profile. The information received from this profile is sent to the RS232 port.

B. Modes of operation of the demonstrator

With the three previous modes of operation of the B2EN modules, one can settle different modes of operation of the demonstrator.

The two B2EN modules can operate together, using one of them to send an ASCII character when a button is pressed, that is displayed on the other B2EN module. To do this, one of the B2EN must be in mode 1 and the other in mode 2.

Another demonstration consists in emulating a RS232 cable connection, in which a B2EN module is connected to each of the devices formerly connected through a cable. In this case both must operate in mode 3.

The other mode of operation is to substitute one of he modules by a Bluetooth enabled PC or PDA. In this case the PC/PDA will receive the ASCII character sent by the B2EN module, or send a character that will be displayed on the B2EN display. For this demonstration the B2EN module must be programmed in Mode 1 and 2 respectively.

The other possibility is to use the B2EN configured as a Serial Port emulation. Mode 3 is then required. The PC/PDA is able to access the device connected to the B2EN module as if it was connected to it with a cable. When using a PC/PDA most of the software developed to work with a hardware UART can be used to conduct the tests.

B. Configuration of the B2EN modules

When configuring the operation mode, it is also necessary to configure the Bluetooth role of the B2EN module and choose the ASCII character that will be used in the paring process as part of the pin. It must also be chosen which, from the communicating devices, will be acting as Master or Slave of the Bluetoth piconet (see section 2).

The configuration process is as follow (see figure 5 to obtain the buttons' position):

Press and release, simultaneously the three buttons <Menu>, <+> and <->.

The display will show the "3" symbol.

Use the <+> and <-> buttons to configure the mode of operation of the B2EN module, according to the previous description.

Press and release the <Main> button.

The display will show the "M" symbol.

Use the <+> and <-> buttons to configure the Bluetooth role of the B2EN module as Master or Slave.

Press and release the <Main> button.

The display will show the "0" symbol.

Use the <+> and <-> buttons to choose the ASCII character that will be used as part of the pin during the pairing process. The possible values are "0 / 1/ 2 / 3 / 4 / 5 / 6 / 7 / 8 / 9 / A / b / C / d / E / F".

Press and release the <Main> button.

Press and release the <Reset> button.

The module will now start working.

D. Operation procedure of the B2EN modules

After the configuration process takes place and before a connection can be established the module must be paired with another device. The pairing is used in order to create a common link key. The pin code used in this process is "pass_X", where "X" is the ASCII character selected during configuration. After a successful pairing the address and link key associated with the paired peer device will be stored and the process will not be repeated until a new configurations takes place. The pairing procedure is executed automatically by the B2EN module software.

Once the device has been paired, a connection can be established between the two devices to implement the desired operation mode. This will be also done automatically by the B2EN module software.

After a configuration, the ASCII character chosen will flash in the display while the pairing takes place and while there isn't a connection established. After a successful pairing the relevant information will be saved. Then a connection establishment is tried. In the future, after any "power cycle" or reset, the modules will automatically establish a connection with the device with which it was paired.

When there is a connection established it will be shown in the display a lighted character that can be the character / sent or received by the module (in mode 1 and 2) or the pin code ASCII character (in mode 3).

V. IMPLEMENTATION DETAILS

A. Firmware choice

To this application the best suited Bluetooth profile seams to be the serial port profile, because it is intended to transmit a few bytes or a stream of bytes. This is the same that happens in the case of using a serial port and cable.

According with the serial port profile the layers and entities in this profile are the ones shown in figure 8.

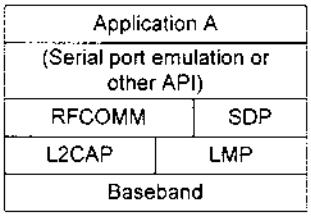


Figure 8 – Protocol model

The Baseband, LMP and L2CAP are the Bluetooth protocols correspondent to the OSI layers 1 and 2. RFCOMM is the Bluetooth adaptation of GSM TS 07.10, providing a transport protocol for serial port emulation. SDP is the Bluetooth Service Discovery Protocol.

The port emulation layer shown is the entity emulating the serial port, or providing an API to applications.

The better way to obtain a fully integrated system is to use a RFCOMM firmware module and to develop the rest of the stack/application in BlueLab [15]. In terms of hardware the two types of modules, HCI and RFCOMM are similar. The difference is in the firmware that they use. In the first type (called by CSR the "HCI firmware build", i.e., the ensemble compiled and ready to use) the module provides the layers up to, and including, the Baseband and LMP of the simplified stack schematic, while the "RFCOMM firmware build" includes also the L2CAP, SDP and RFCOMM layers.

B. Bluetooth Stack

The Blueetoth Stack is implemented using a scheduler to co-ordinate the stack layers execution. The scheduler provides a single-threaded co-operative non-preemptive multi-tasking environment. A direct consequence of the use of this simple scheduler is that there can be no blocking calls. A message passing method is used where a message is sent to a particular task via the scheduler, using a queue identifier as the destination. The scheduler executes the tasks that have messages pending. It is up to the software module to pull the message or messages from the queue and to act on them. The result of this is that the Bluetooth protocol stack becomes message driven.

C. Application

CSR has provided in the BlueLab development tool some libraries that sit on top of the Bluetooth Stack or use the module hardware to provide useful results. The most important are the Event library, the Persistent Store library, the Stream library, the Message and Scheduler libraries, the Buttons library and the Connection Manager library.

This last one is very useful to implement the Serial Port Profile because it is intended to allow simple, low-traffic RFCOMM connections between a pair of devices. It uses MessageQueue 0 for incoming messages and posts outgoing messages to MessageQueue 1.

The MessagesQueues are a way of passing messages between tasks/libraries. The tasks that have pending messages will be executed.

In what concerns our application, it consists essentially of a main program and a main task. The main program performs some initialization and launches the scheduler supplied by the tool suite. The events related with Bluetooth are handled by task 1. Task 1 executes whenever there are messages in MessageQueue 1. These messages are posted by the Connection Manager. The task produces messages posted to MessageQueue 0, thus triggering the execution of the Connection Manager. These messages are often requests that are answered with Indication and Confirm messages.

The message sequence passed between the Connection Manager and an application to execute the basic functions related to the connection establishment are the following:

Initialization of the Connection Manager:

- CM_INIT_REQ (application to connection manager)
- CM_INIT_CFM (connection manger to application)
- CM_OPEN_REQ (application to connection manager)
- CM_OPEN_CFM (connection manger to application)

Starts the Connection Manager and drives it to a pre-established state.

Inquiry:

- CM_INQUIRT_REQ (application to connection manager)
- CM_INQUIRY_RESULT_IND (connection manger to application)
- CM_INQUIRY_COMPLETE_CFM (connection manger to application)

Used to discover devices within the range.

Pairing as master or slave:

- CM_PAIR_REQ as master/slave (application to connection manager)
- CM_PIN_CODE_REQ (connection manger to application)
- CM_PIN_CODE_RES (application to connection manager)
- CM_PAIR_CFM pairing not finished (connection manger to application); the link key is received if it is the master
- CM_PAIR_CFM pairing complete (connection manger to application); the link key is received if it is a slave

Executes the pairing with a device to obtain the link key, being master or slave of the piconet

Add paired device to security manager, to avoid pairing procedure

- CM_ADD_SM_DEVICE_REQ (application to connection manager)

This is done when the device is starting operation with available information from a previous pairing (link key and Bluetooth address already available).

Connect as master

- CM_CONNECT_AS_MASTER_REQ (application to connection manager)
- CM_CONNECT_CFM connection complete (connection manger to application)

Connect as slave

- CM_CONNECT_AS_SLAVE_REQ (application to connection manager)
- CM_CONNECT_CFM connection complete (connection manger to application)

With this basic steps and a few more code the application that runs on the B2EN modules was created. In figure 9 it is shown the state machine that establishes the Bluetooth connection.

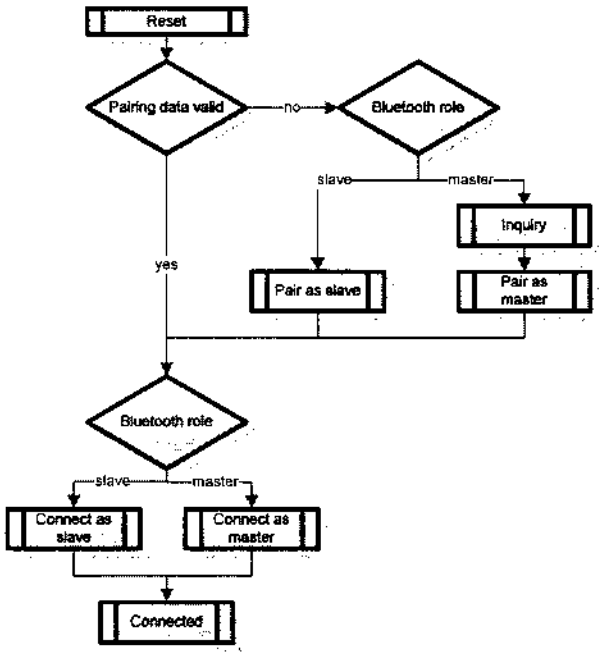


Figure 9 -- Bluetooth application state machine

As it can been seen, after a device is connected, if it has valid pairing information it will immediately establish a connection with the device with which it was paired. If it hasn't been already paired the slave will enter a state where it can receive pair requests, while the master will discover the available devices in its range and try to pair with them. During the pairing a Pin Code is used. This code is being used to limit the number of devices allowed for pairing. If both devices use the same pin the pairing will be done. In the other cases, the slave will keep accepting pairing requests and the master will return to execute a new inquiry to try to find a device that is using the same Pin Code.

Once the connection is established the functionalities of the Stream library are used to send and receive information through the created RFCOMM connection. When the device is working as RS232 port emulation the UART Source Stream is connected to the RFCOMM Sink Stream and the UART Sink Stream is connected to the RFCOMM Source Stream.

When the device is in mode 1 it uses an event generated when data arrives to a source to display the received data in the display. When it is in mode 2 it uses another event that is generated when a button is pressed to send the ASCII character to the RFCOMM sink. These functionalities are provided by the Event and Button libraries, jointly with the Stream and Scheduler libraries.

The configuration procedure is implemented using the Button library event and a state machine to drive the user through the configuration options. It uses the Persistent Store library to save the configurations in non volatile memory. This same library is also used after a successful pairing to save the link key, avoiding the need to repeat the pairing procedure in the future, thus accelerating the connections.

D. Development process

The code is written in C and is compiled with the compiler and linked with the libraries provided by CSR in the BlueLab development tool. The compiler and linker run under CygWin, a LINUX like emulator. The source code of some of these libraries is also available, so the user can modify it if necessary.

During the development the code can be compiled to a form suitable to be loaded in a simulator that is provided with the tools suite. This is done for debugging purposes. A Bluetooth module connected to the PC provides the lower level functionalities of the simulator.

After the code is in a more advanced development stage it can be downloaded to the target modules through a SPI interface that connects to the parallel port of the PC.

This SPI interface can also be used to work with the Persistent Store Keys that hold the configuration and that are used by the hardware and Bluetooth stack.

VI. CONCLUSIONS

In this paper a demonstrator of the operation of Bluetooth modules developed at the Electronic Systems Laboratory of IEETA was presented. This demonstrator can be used to show Bluetooth connections between two Bluetooth modules or between the modules and a Personal Computer. Besides the visible demonstration using buttons and 7-segment displays, the demonstrator provides a functionality similar to the Serial Port Profile. The demonstrator Bluetooth modules, called B2EN – Bluetooth Basic Embedded Nodes, were developed using Airlogic modules and the CSR BlueLab development tool suite. A short overview of the development process was also presented in the paper.

REFERENCES

[1] Bluetooth SIG, "Specification of the Bluetooth System 1.0", July 1999.

[2] Bluetooth SIG, "Specification of the Bluetooth System 1.0B", December 1999.

[3] Bluetooth SIG, "Specification of the Bluetooth System 1.1", Specification Volume 1, February 2001.

[4] IEEE Standard for Information technology—Telecommunications and information exchange Between systems—Local and metropolitan area networks—Specific requirements, "802.15.1 -Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Wireless Personal Area Networks (WPANs)", June 2002.

[5] Bluetooth SIG, "Bluetooth 1.2 Core Specification", November 2003.

[6] Bluetooth SIG, "Bluetooth 2.0 Core Specification", November 2004.

[7] ambridge Silicon Radio, http://www.csr.com, April 2005.

[8] Broadcom, http://www.broadcom.com, April 2005.

[9] RF Micro Devices, http://www.rfmd.com, April 2005.

[10] Chatschik Bisdikian, "An Overview of the Bluetooth Wireless Technology", IEEE Communications Magazine, December 2001, pp. 86-94.

[11] Jaap C. Haartsen, "The Bluetooth Radio System", IEEE Personal Communications, February 2000, pp. 28-36.

[12] McDermott-Wells, P., "What is Bluetooth?", Potentials, IEEE ,Volume 23, Issue 5, Dec. 2004-Jan. 2005, pp. 33-35.

[13] Robert Morrow, "Bluetooth Operation and Use", McGraw-Hill, 2002.

[14] European Telecommunications Standards Institute (ETSI), "3GPP TS 07.10 version 7.2.0 - Digital cellular telecommunications system (Phase 2+) Terminal Equipment to Mobile Station (TE-MS) multiplexer protocol", 1998.

WEB PAGES REFERENCES

[13] Airlogic Co. Ltd, http://www.airlogic.co.kr/

[14] Cambridge Silicon Radio, CSR http://www.csr.com/home.htm

[15] BlueLab SDK for single-chip BlueCore Applications http://www.csr.com/development/bluelab.htm

[16] BlueCore2, http://www.csr.com/products/bc2range.htm

QoS-aware Fast Handover Optimization Supported by Multicast Networks

Nuno João Sénica, Rui L. Aguiar, Susana Sargento

Abstract— This paper presents three different solutions for handover optimization covering three different scenarios. The first scenario suits the needs of an operator driven network with no degree of liberty on the choice of the new Access Router (AR) by the terminal. An higher degree of liberty is achieved in the second proposed solution, where a supporting multicast network grants the non predictability of the target AR. The multicast network allows the reduction of the bandwidth usage inside the operator network assuring resource optimization, and the delivery of the packets to the surrounding ARs, and thus to the roaming terminal. Nevertheless, these two methods (which are operator driven) depend on an entity in the network for handover permission and control. To avoid this in a high mobility network, we propose a third solution where there is no admission control and always assures available resources in the surrounding *ARs*.

***Index Terms*— Fast mobility, QoS, multicast**

I INTRODUCTION

Nowadays, operators feel the pressure to provide the best service they can to their customers. The deployment of heterogeneous networks is thus a pressing reality to them. Mobility is a 'must' in those heterogeneous networks and as a consequence of its heterogeneity, mobile nodes can potentially roam between different types of access network (e.g. WLAN, 3G).

The Internet Protocol version 6 (IPv6) has a vital role in these heterogeneous environments for data traffic as well as for multimedia applications, providing a convergence layer for seamless mobility, Quality of Service and multicast. IPv6 already includes basic mobility support. However, in order to achieve fast, efficient and seamless mobility it is required that no packet loss is felt, no interruption or degradation should be noticed by the user or its corresponding nodes. With the growing number of wireless users, scalability is also an issue when designing new architectures since a large number of handovers may potentially occur at the same time.

With all these requirements in mind, we present three fast handover architectures. All three architectures aim to provide seamless handovers although they have different applications and characteristics. The first two proposals are adequate for operator-driven networks, combining the integration of mobility with Quality of Service, granting seamless mobility with QoS support, with multicast networks in the case of the second architecture. The third proposed architecture envisions a high mobility network supported by a multicast network, also to achieve seamless mobility.

The paper is organized in the following way. In section II we summarize the mobility, Quality of Service and multicast solutions considered and its integration. Section III details the considered architectures and its qualitative evaluation. Finally in Section IV we present our conclusions.

II BACKGROUND

This work bring together Mobility solutions associated to QoS, using Multicast Technologies as a supporting tool.

Mobility Solutions

The Mobile Internet Protocol version 6 (MIPv6) [1] is the current IETF standard to provide global mobility management and to enable mobile nodes (MN) to roam across different networks, maintaining its reachability to and from other nodes in the Internet. MIPv6 creates a new care-of-address (CoA) that represents the mobile node's new location and advertises this to its correspondent nodes and to a mobility manager (Home Agent, HA) in the home network. To support mobile Internet users, a MN has thus two IP addresses assigned, one fixed (the "identification" home address), and the other changing (the "topologically correct" CoA). Even if MIPv6 potentially enables mobile Internet users to be always reachable regardless of the specific access network technology, increasing multimedia demands from mobile users highlighted MIPv6 shortcomings. Real time audio/video applications underline the need to have in place mechanisms minimizing the large handover latency and service degradation (eg. packet loss) usually associated with MIPv6. In [2] different micro mobility management schemes, such as Cellular IP [3] and Hierarchical Mobile IP [4], offering fast and seamless local mobility are discussed and compared. This comparison is purely based on the evaluation of local mobility management schemes taking into account handover latency, packet loss and scalability issues without integration concerns. Other mobility mechanisms enhancing Mobile IPv6 to account for performance issues are further defined in the IETF, such as Fast Handover [5], recently announced as experimental RFC. The Fast Handover (FHO) proposal, which represents the initial influence for this work, is based on the "make before break" approach, where the terminal signals its handover with the new network using its current connection through the old network. Moreover, during

handover, the packets are sent to the mobile node both via the old and the new network to prevent the packet loss during the handover period. One other issues, is that in next generation networks, fast mobility has to be considered along with QoS profiles. In [6], an enhanced fast handover stack was designed and implemented as an extension to MIPv6, also exploiting the FHO basic ideas.

Integrating Fast-Handovers with a QoS Subsystem

The basic fast handover signalling [5] has been extended to support the integration of QoS, with a QoS Broker, a resource manager. The adopted "make before break" philosophy allows to prepare an handover by informing the new point of attachment in the network previous to the handover. In that process, inter access router communication enables the transfer of user related information such as security information and user profile. Achieved results show that handovers do not last for more than 30 msec, assuming idealized QoS components (e.g. only local computation), and that there is no performance degradation during handover in real-time UDP traffic and TCP data transfer [7]. The complete handover process, from the moment the terminal decides to handover until it performs the binding in the new network, including real QoS delays, does not last more than 130 msec [8], a major improvement over standard MIPv6. Other similar proposals have been discussed in the literature. As an example, [9] presents an end-to-end QoS architecture that enables roaming services over heterogeneous wireless access networks. The proposed scheme is also based on a resource manager approach where each autonomous system implements a Domain Resource Manager. The authors present an integrated state model aiming at run time switching between different kinds of handovers in case of failure while preserving reservations. Several types of handovers are here supported: inter- and intra-domain, vertical and horizontal handover, but no mechanism is provided to achieve no packet loss.

Multicast IPv6

IPv6 was developed from the start taking in consideration IP multicast. IPv6 multicast protocols evolved from their IPv4 counterparts, creating a solid base for the tight deployment of IPv6 and IP multicast.

IPv6 multicast is supported by several fields and protocols: a 128 bits group address space, a scope identifier for domain control of the multicast group, a protocol-independent routing protocol, designated by Protocol Independent Multicast (PIM) [10], and group membership mechanisms, designated by Multicast Listener Discovery (MLD) [11]. Group membership in IPv6 multicast is handled by MLD. Its purpose is to enable terminals to communicate the multicast group they which to subscribe to the multicast enabled router. Periodically, the router queries the terminal on the groups it wishes to maintain subscription.

Multicast routing uses PIM and its variants: Sparse Mode (SM) and Source Specific Multicast (SSM). PIM is used to construct the multicast tree used to forward the multicast packets from the source to the terminals. These trees can be based in two different approaches. PIM-SM employs a special configured router, denoted by Rendezvous Point (RP), that serves as a meeting (common) point for multicast senders and listeners. Leaf routers that detect multicast listeners via MLD generated join messages and send them in unicast to the RP's. The PIM-SM also supports the Any-Source Multicast (ASM) model. The ASM model is appropriate for multicast applications such as multiparty videoconferencing, in which multiple sources transmit to the same group

PIM-SSM only supports source-routed deliver trees, and therefore does not use or require an RP. The leaf router learns, via MLDv2 [12], the IPv6 multicast group address and the sender's IPv6 unicast address. This combination of source unicast and group multicast addresses *(S,G)* identifies a channel in the SSM model. Broadcast media applications are therefore natively supported by the SSM model.

III CONSIDERED ARCHITECTURES

In this section we present three proposed architectures, designed to achieve fast mobility and minimization of the real-time session degradation. The first proposal considers a fast mobility scheme enhanced from FMIPv6 extension used in the Daidalos IST Project [13], with mobile terminal and network initiated handover. The second proposal considers the previous enhanced FMIPv6 proposal with multicast supporting networks to enhance the handover efficiency. Finally, a novel mechanism is proposed that uses a supporting multicast network to guarantee an "always on" paradigm on fast moving mobile nodes.

Fast Mobility in the Daidalos Project

Envisioning environments with high level of mobility requires the minimization of the overhead required for signalling and focus on access resource control (typically considering the Diffserv model). Besides being integrated with QoS, the fast handover approach presented in this paper extends the previous ones in its ability: (1) to be independent on the mobility protocol in use, even if it is implemented with basis on the FHO; (2) to address both mobile initiated and network initiated handover; and (3) to potentiate the existence of interface selection entities through the information provided by existing network discovery mechanisms such as [14] and an intelligent decision module in the mobile terminal.

The QoS reference architecture is *DiffServ* based and relies on a central resource management entity, the QoS Broker (QoSB). It performs admission control and manages network resources, controlling the mobile node's services, the network routers and its reservations. The QoSB is also responsible for handover authorization, verifying if the

node can use the requested resources on the new link. A Performance Manager module located in the QoSB gathers reports (link availability, signal measurements, etc.) from Performance Attendants located in every access router (AR). By means of this information it computes an algorithm in order to optimize radio link resources, and determines if and which mobile node(s) need(s) to change their current point of attachment. Several mobile nodes then receive a notification from the QoSB to change their point of attachment: this process is denoted as **Network Initiated Handover**. Thus, the network can impose an handover because of network performance and geographical mobility. The former aspect mainly deals with resource optimization and network load balancing. The latter addresses the connectivity problems caused by signal level degradation. Similar mechanisms apply to nowadays 2G/3G networks.

The mechanism so far described mainly considers access network specific operations. However, the proposed architecture takes into account user preferences and requirements, by providing an interface selection scheme able to guarantee communication capabilities according to the "always best connection" paradigm. Thus, the MN can independently decide to request an handover taking into account terminal related conditions (e.g. wireless signal level fast degradation) and user preferences (e.g. a better and cheaper available connection). This operation is identified as **Mobile Terminal Initiated Handover**. Notice however that the network, more specifically, the QoSBs in charge of the old and new access networks, needs to finally authorize this mobile terminal request, and that the handover is only performed after this authorization. In the next sub-sections we detail both the mobile terminal and network initiated handovers.

IV MOBILE TERMINAL INITIATED HANDOVER

In the first phase of the handover process, the MN needs to bootstrap a handover preparation mechanism to discover available candidate access routers. For this purpose, the CARD protocol is used - CARD components are located on the access routers and MNs. The fast handover preparation and execution processes are depicted in Figure 1. Upon receiving information on the available ARs (eventually offering access in different technologies), a MN can decide to roam to another AR, e.g. because of user preferences, by sending a *HandoverRequest* message (message 1) to its current AR containing an ordered list (up to three), of selected candidate ARs. This message has a flag indicating if the Handover (HO) is imminent (e.g. lost of communication is imminent) or not. Upon the reception of this message, the AR sends a request for handover approval to the QoSB (*HandoverRequest* – message 2). Thus, the QoSB verifies whether the resources are available on the indicated ARs and answers (authorizes) with the first occurrence of the list matching the user's current services requirements. For supporting this authorization, the QoSB sends a *HandoverDecision* message (3, 4) to both the old and new ARs (oAR and nAR). The nAR books the reservations and starts to buffer the packets sent to the new nCoA. The oAR processes the *HandoverDecision*, starts the duplication of the streams directed to the old mobile node's location and triggers the Context Transfer. Finally, it informs the MN that it can now move to the selected AR. As soon as the MN receives the *HandoverResponse* (5) which contains the decision, it performs the necessary internal checks and sends a *Fast Binding Update* (FBU) message to the oAR (6), which then is reported to the QoSB (7). Then, the MN configures the layer 2 connection on the new link (e.g. layer 2 handover) and sends a *Fast Neighbor Advertisement* (FNA) (8) message in order to populate nAR neighbor cache where buffered packets may be already waiting to be delivered. Finally, the nAR sends to the QoSB the information about the successful handover (9). This information is then transferred to the oAR (10) to inform that the handover already occurred. At this time, the packet duplication process is stopped, and the oAR reports to the QoSB (11) that every information about this terminal was deleted, handing over the MN control to the nAR.

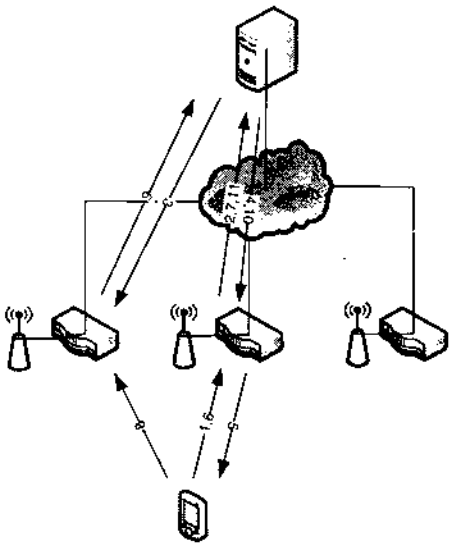


Figure 1 - Fast Mobility Scheme in Daidalos Project

Notice that in the case of the old and new ARs belonging into different access networks, when the QoSB in the old network receives the HandoverRequest message (2), it needs to contact the QoSB in the new network to ask for available resources and to transfer the context related to users, sessions and QoS. The new QoSB answers with the resources availability and the old QoSB can then send the HandoverDecision message to the old AR.

V NETWORK INITIATED HANDOVER

In this case, no preparation phase is required. The network is both the decision and selection point. The QoSB communicates to the AR by means of a *HandoverDecision* message which MN(s) have to roam. Thus, the AR sends a *HandoverResponse* with a special flag set. This flag

indicates that the message is unsolicited. Upon the reception of this datagram, the MN must perform the necessary steps to attach to the new target AR, since it knows that its current point of attachment will not be available in the near future. The MN, in case the QoSB indicated a candidate no longer available (i.e. out of the coverage area), can still request an abort to the handover procedure by sending the FBU with a negative acknowledge. The roaming steps are the same mentioned above.

Fast mobility supported by a Multicast Network with QoS integration

Multicast networks are the best choice to transport the same traffic inside a network without using duplication mechanisms. With the assumption of a multicast network and the previous fast mobility process in mind, an integrated architecture was designed. The integration of these two techniques includes an extra step in the target selection. The MN does not need to rely on additional protocols to discover surrounding networks, which is a time consuming operation and may cause an interruption on the current connectivity (since it has to disconnect, survey the wireless channels and connect again). This operation is done by the network: since we are considering handovers inside the same domain, the network administrator has the complete knowledge of the network topology. Using this knowledge, the administrator can configure the QoSB with the network topology. The QoSB can then select the proper surrounding AR when a *HandoverRequest* is made, in mobile node and network initiated handovers.

In order to guarantee that no packet is lost in this process, a *any source multicast* network is established using the known network topology; this *any source multicast* network is formed by each AR (as source) and its surrounding neighbours at the network's boot up.

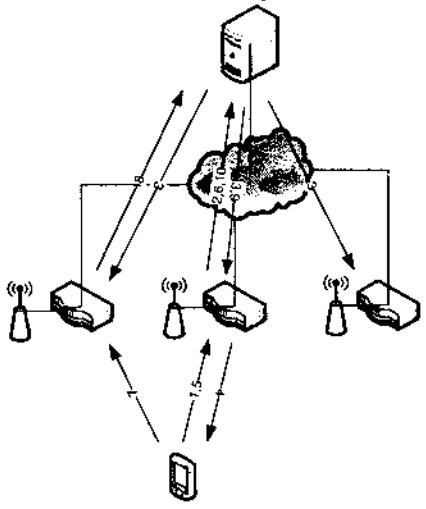


Figure 2 - Fast Mobility Scheme supported by a Multicast network integrated with a QoS System

This fast mobility process is depicted in Figure 2. The MN sensing lower signal in the current AR, performs a *HandoverRequest* (message 1) directed to the AR which is then forwarded to the QoSB (2). The QoSB looks up the MN surrounding networks, checks which networks can handle the current MN's connections and answers back with the possible targets (3). These targets are then informed of the possible handover, and that they are candidate targets and need to be prepared to handle the MN (3).

The current AR, upon the reception of this response from the QoSB, forwards the information to the MN (4). If the handover is allowed, it starts to intercept the traffic directed to the MN inserting it in the previously established *any source multicast* network. At this point, all the surrounding ARs receive the traffic directed to the MN, buffering it for delivery when the MN attaches or until they are instructed by the QoSB that the handover procedure is complete. At this point, the MN can now freely move to any of the candidates listed by the QoSB. Before leaving its current network the MN sends a *FastBindingUpdate* to the AR (5) which is then reported back to the QoSB (6). As soon as it is attached to a network it sends a *FastNeighbourAdvertisement* (7) and the AR starts delivering the packets to the MN which also needs to send a *Binding Update* to its corresponding nodes (CN) (not depicted for readability issues). The CNs then send the *Binding Update Acknowledgment* to the MN. At this moment the AR triggers the information of reception of the MN to the QoSB (8). The QoSB forward this message to the previous AR (9) in order to inform if the handover was successful and, if so, to stop inserting any remaining traffic in the *any source multicast* network. The previous AR reports the successful handover back to the QoSB (10). Each of the ARs informed of the handover have a handover time frame for its success; if the FNA message does not arrive in that time frame, the buffered packets are discarded and they start discarding all the incoming traffic directed to the MN.

After all these steps, the MN is directly communicating with its CNs with no interruption of the current communication.

In a Network Initiated Handover scenario, the MN receives the order to move to one of the candidate targets following the previously described procedure.

Fast mobility supported by a Multicast Network

The fast mobility mechanism presented in this section is addressed to a fast mobility network, where MNs are always moving with a very high probability to be in a low signal coverage or in overlapping areas. With these requirements, a fast mobility scenario without any intervention of bandwidth management mechanisms was designed. This mechanism is presented in Figure 3.

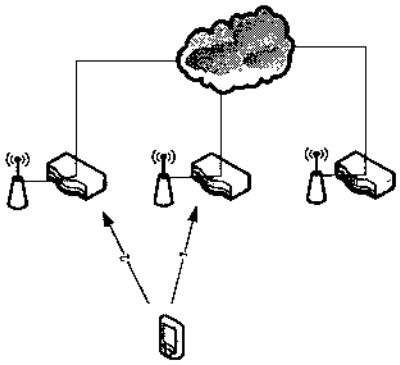


Figure 3 - Fast Mobility Scheme supported by a Multicast network

As in the previous presented solution, there is an *any source multicast* network previously established between each AR and its neighbours. All the traffic directed to a MN is intercepted and inserted in the *multicast* network corresponding to that AR. At this point, all its neighbours receive the traffic, which has a small lifetime in the surrounding ARs buffers. For a small period of time, the ARs keep the traffic in order to guarantee the delivery to the roaming MN as soon as it attaches and signals the attachment. Periodically, the MN sends *Keep-Alive* messages to its current AR (message 1). These messages signal the AR that the MN is still connected to that AR; as long as the AR is receiving this signal, it inserts the traffic into the multicast network. When the MN moves, it senses new ARs. To attach to a new AR, the MN signals the attachment with a *Keep-Alive* message (2). It also sends the *BindingUpdate* message to the CNs. When a AR receives the *Keep-Alive* message, it starts to insert the unicast traffic directed to the MN into its *any source multicast* group, preparing a future handover of the MN. After a time out of 3 *Keep-Alive* messages, the previous AR stops introducing the traffic into the multicast network. At this stage, the handover procedure is concluded.

Architectures Evaluation

These three fast handover mechanisms have advantages and disadvantages, which make them best suited to specific situations and/or scenarios.

The first two methods are very similar. The differences between both rely on the way how the packets get to the new ARs and the selection of the new target AR. Comparing the first two methods, it is possible to find some similarities, as both depend on a central entity to control the QoS, which is also responsible for the admission control and therefore for the handover authorization. One of the large advantage of having an *any source multicast* network to support duplicated packets is the previous knowledge of the MNs' surrounding ARs. Also, the multicast groups already assigned allows the MN to move to a finite set of those ARs in the neighbouring, deciding according both to signal and user preferences. Although this is an advantage for the classic fixed network where the ARs are fixed for a long period of time, it is not a good solution for moving ARs, since it is required to continuously update the topology in the QoSB (create the *any source multicast* networks requires some time to set up and balance). This problem is not present in the first presented solution, since the MN communicates which ARs it can attach to, and the QoSB decides which of the ARs can handle the MN. However, this process limits the MN choice of ARs, and it is subject to problems in highly dynamic networks where the signal level can change very fast.

Due to the existence of a previous multicast group including the neighbouring routers, the second approach has a large advantage in the handover time. These source/group multicast networks may also be controlled by the QoSB, since it can inform each AR of its multicast network and its surrounding ARs. With this information, the AR can start the join process and establish the multicast networks at boot up (or when it is informed by the QoSB).

In terms of overhead, the second approach has a significant signaling overhead in the wired network. This is due to the existence of a control entity. However, the overhead in the wireless link is low, since the traffic is only inserted in the multicast group upon handover request.

The third approach does not contain a control entity, and therefore, there is no access control and no QoS guarantees in the new network, both for the new flow and for the ones already present in the network. Also, it requires the complete knowledge of the network topology in order to establish the multicast networks. However, this procedure requires very small signalling overhead, and provides a handover really fast without packets loss and additional signalling. In terms of data overhead, all the ARs belonged to the multicast group receive the same data stream, which increases the resource usage in the core networks. However, the core network is usually not the bottleneck compared to wireless link. Moreover, this is the only way to grant a continuous stream to wandering MNs. This scenario is the best suited one for very large mobility scenarios.

VI CONCLUSION AND FUTURE WORK

This paper presented three different solutions for handover optimization covering three different scenarios. The first scenario suits the needs of an operator driven network with no degree of liberty on the choice of the new AR by the MN. The MN always depends on the QoSB decision on the next network to move. This degree of liberty is achieved in the second proposed solution, where a supporting multicast network grants the non predictability of the target AR (in this case a set of neighbouring ARs are prepared to receive the MN). The multicast network allows the reduction of the bandwidth usage inside the operator network assuring the resource optimization, and the delivery of the packets to the surrounding ARs, and thus to the roaming MN. Nevertheless, these two methods (which are operator driven) depend on an entity in the network for handover permission and control. To avoid this in a high mobility network, we proposed a third solution where there is no

admission control and where there are always available resources in all surrounding ARs.

As future work, it is planned to evaluate in details these three solutions within different scenarios, in order to understand which suits better in a particular scenario. Extensions and updates to the presented architectures will be performed as results of the simulation results.

REFERENCES

[1] D. JOHNSON, C. PERKINS. *MOBILITY SUPPORT IN IPV6*.JUNE 2004, RFC 3775

[2] A. Campbell, J. Gomez, S. Kim, and C. Wan. *Comparison of IP micro-mobility protocols*. In IEEE Wireless Communications, vol 9, pages 72–82, February 2002.

[3] A. Campbell et al., *Cellular IP*. Internet draft, draftietf-mobileip-cellularip-00, work in progress, Dec. 1999.

[4] Hesham Soliman et al. *Hierarchical mobile IPv6 mobility management* (hmipv6), June 2003. http://www.ietf.org/internet-drafts/draft-ietf-mobileiphmipv6-04.txt.

[5] Rajeev Koodli (ed). *Fast handovers for mobile IPv6*, July 2004. http://www.ietf.org/internet-drafts/draft-ietf-mipshop-fast-mipv6-03.txt.

[6] V. Marques, R. L. Aguiar et al. *An IP-based QoS architecture for 4G operator scenarios*. IEEE Wireless Communications, June 2003.

[7] T. Melia, R. Schmitz, and T. Bohert. *TCP and UDP performance measurements in presence of fast handovers in an ipv6-based mobility environment*. In 19th World Telecommunications Congress (12-15 September 2004, Seoul, Korea), September 2004.

[8] V. Marques. *Multimedia services for heterogeneous networks*. In Phd Thesis, October 2004 Universidade de Aveiro.

[8] D. Estrin, D. Farinacci, A. Helmy, D. Thaler, S. Deering, M. Handley, V. Jacobson, C. Liu, P. Sharma, L. Wei, *Protocol Independent Multicast – Sparse Mode (PIM-SM): Protocol Specification*, RFC 2362

[9] R. Bless, J. Hillebrand, C. Prehofer, and M. Zitterbart. *Quality-of-service signaling for next generation ip-based mobile networks*. volume 42 of Communications Magazine, pages 72–79. IEEE, June 2004.

[10] S. Deering, W. Fenner, B. Haberman, Multicast Listener Discovery (MLD) for IPv6, RFC 2710

[11] R. Vida (Ed), L. Costa (Ed.), Multicast Listener Discovery 2 (MLDv2) for IPv6, RFC 3810

[12] The Daidalos Consortium. November 2003. http://www.ist-daidalos.org

[13] Marco Liebsch (ed). *Candidate access router discovery*, Sep 2004. //www.ietf.org/internet-drafts/draft-ietf-seamoby-card-protocol-08.txt.

Modelização da dispersão de um compensador dinâmico de dispersão cromática baseado em redes de Bragg de período variável gravadas em fibra óptica

B. Neto[1], M. J. N. Lima, A. L. J. Teixeira, R. N. Nogueira[1], J. L. Pinto[1], J. R. F da Rocha, P. André[1]

[1] Departamento de Física
Universidade de Aveiro, 3810-193 Aveiro

Resumo **– Reportamos um procedimento para a realização de uma compensação dinâmica da dispersão cromática. O compensador baseia-se em redes de Bragg de período variável gravadas em fibra óptica (CFBG), sendo que o controlo de um gradiente de temperatura aplicado na CFBG é usado para variar a sua dispersão. Apresentamos um modelo termodinâmico que nos permite inferir a temperatura local na CFBG e posteriormente a sua dispersão. Os resultados foram validados com dados experimentais**

Abstract **–** *We report a method to realize tunable chromatic dispersion compensation based on chirped fiber Bragg gratings (CFBG). The changes on the CFBG dispersion are achieved by controlling a temperature gradient applied to the grating. A thermodynamic model is presented, enabling us to simulate the temperature distribution along the grating length. Furthermore, a theoretical approach was developed to find the dispersion and the dispersion slope for several values of temperature gradients. The model results are in good agreement with the experimental data.*

I. INTRODUÇÃO

A dispersão cromática é um efeito linear que se traduz no alargamento temporal de impulsos que se propagam ao longo de uma fibra óptica, sendo por isso responsável pela degradação da qualidade da informação transportada. É de referir que este efeito é particularmente relevante quando se utilizam elevados ritmos de transmissão (acima de 10 Gbits/s), fomentando assim o desenvolvimento de dispositivos ópticos compensadores. A necessidade de compensação dinâmica é requerida sempre que são observadas alterações na dispersão cromática de um sistema de comunicações ópticas, devidas a perturbações ambientais como a temperatura [1], ou à reconfiguração de percursos ópticos.

De entre as várias possibilidades tecnológicas de compensar no domínio óptico a dispersão cromática, destacamos as redes de Bragg de período variável gravadas em fibra óptica (CFBG) pelas vantagens apresentadas, nomeadamente o baixo custo de implementação, as dimensões reduzidas e a elevada flexibilidade quando comparadas com outros métodos. A aplicação de gradientes de temperatura ao longo da CFBG pode ser utilizado como elemento de sintonia da dispersão. Neste artigo, procede-se ao estudo dos processos de transferências de calor envolvidos numa CFBG em que a temperatura nas suas extremidades é mantida constante mas cuja superfície lateral está em contacto com o exterior, com o intuito de obter uma distribuição espacial de temperatura e assim modelizar a dispersão do dispositivo.

Na secção II descrevem-se as CFBG, enfatizando as suas características espectrais e a dispersão. Na secção III é apresentado o dispositivo compensador, assim como, os modelos que permitem inferir a distribuição de temperatura e a dispersão. Na secção IV mostram-se resultados obtidos experimentalmente que são posteriormente comparados com resultados teóricos obtidos com o modelo aqui reportado. Finalmente, na secção V apresentam-se as conclusões finais do artigo.

II. REDES DE BRAGG GRAVADAS EM FIBRA ÓPTICA

As redes de Bragg gravadas em fibra óptica consistem numa modulação periódica do índice de refracção do núcleo da fibra, induzida por exposição a um padrão de luz ultravioleta. Quando se verifica a condição de ressonância, um máximo de reflectividade é observado para um determinado comprimento de onda (que satisfaz a condição de Bragg). O valor do comprimento de onda de Bragg fica estabelecido pelo índice de refracção efectivo da rede e pelo período espacial de modulação do índice [2]:

$$\lambda_B = 2n_{eff}\Lambda \qquad (1)$$

Numa rede de Bragg de período variável (CFBG), o período óptico $n_{eff}\Lambda$ não é constante ao longo do seu comprimento. Uma rede deste tipo obtém-se por variação

do índice de refracção efectivo com o comprimento $n_{eff}(z)$, ou por variação do período espacial da amplitude de modulação do índice $\Lambda(z)$, ou por variação simultânea de ambas as grandezas. Por razões técnicas é usualmente utilizada a modulação do período espacial da amplitude de modulação.

As CFBG com aperiodicidade linear revelaram elevado potencial na compensação da dispersão cromática uma vez que introduzem um atraso de grupo constante entre componentes espectrais igualmente espaçadas. É possível dimensionar fisicamente este tipo de rede por forma a obter um determinado valor de dispersão (simétrico daquele que apresenta a fibra óptica) e assim proceder a uma recompressão temporal do impulso óptico que sofreu um alargamento temporal na sua propagação.

A dispersão numa rede de Bragg cujo período óptico varia linearmente ao longo do seu comprimento é dada por:

$$D_{FBG} = \frac{2n_{eff}L}{c\Delta\lambda} \tag{2}$$

em que $\Delta\lambda$ é a largura espectral da rede e designa a diferença entre os comprimentos de onda de Bragg reflectidos no início e no fim da rede, L é o comprimento da rede e c é a velocidade de propagação da luz no vazio.

Uma variação linear do período espacial de modulação do índice, resulta numa largura espectral de:

$$\Delta\lambda = 2n_{eff}C_{\Lambda}L \tag{3}$$

onde C_Λ é o coeficiente de aperiodicidade da rede ($\Lambda=\Lambda 0+C_\Lambda z$). O sinal de C_Λ está relacionado com a monotonia do período óptico.

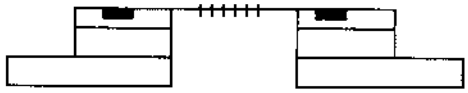
Para utilização na compensação da dispersão cromática das fibras SMF, o sinal óptico deverá entrar na rede pelo extremo onde o período espacial da amplitude de modulação do índice é maior, o que significa que C_Λ terá sinal negativo e consequentemente a dispersão da rede será também negativa.

III. ABORDAGEM CONPECTUAL – MONTAGEM EXPERIMENTAL

Para estudar a compensação dinâmica de dispersão cromática baseada em redes de Bragg, gravou-se uma rede CFBG numa fibra óptica padrão, previamente hidrogenizada, por forma a incrementar a sua fotossensibilidade. A gravação foi realizada utilizando o método da máscara de fase e um laser de iões de Árgon dobrado em frequência (244 nm). A rede gravada apresenta um comprimento de 2.35 cm e um coeficiente de aperiodicidade (C_Λ) de -0.6 nm/cm, estando o espectro de reflexão da mesma, centrado num comprimento de onda de 1546 nm.

A rede foi colocada entre dois suportes de latão espaçados de 3 cm, tal como se mostra na figura 1. Estes suportes encontram-se ligados a módulos termo-eléctricos (TEC), através dos quais se pode controlar a temperatura de cada um dos blocos. Embutidos nos blocos de latão estão colocados *termistors* que permitem controlar a temperatura efectiva nos blocos de latão, providenciando um sinal de realimentação para as fontes de controlo dos TEC. As temperaturas atingidas nos blocos variam entre 0 °C e 80 °C, funcionando os mesmos como reservatórios térmicos.

(a)



(b)

Figura 1- Dispositivo compensador de dispersão cromática. Fotografia (a) e esquema lateral (b) do dispositivo.

□ - suporte de cobre, □ -TEC, ■ - termístor.

A. Modelização dos processos de transferência de calor na CFBG

O dispositivo apresentado pode ser fisicamente modelizado como uma barra cilíndrica de vidro com as bases a temperaturas fixas, T_1 e T_2, mas cuja superfície lateral se encontra em contacto térmico com o exterior à temperatura T_{amb}. Ao longo da barra, há transferência de calor por condução devido ao gradiente de temperaturas estabelecido, mas também trocas com o exterior por convecção e radiação, uma vez que aquela não apresenta isolamento térmico ao longo da superfície lateral. A modelização apresentada pretende determinar a evolução temporal da temperatura local ao longo da fibra tendo em conta a difusão de calor e sem esquecer a ocorrência de mecanismos de trocas de calor com o exterior. A lei empírica que descreve a transferência de calor por condução é dada por [3]:

$$q = -K_{fibra}\frac{\partial T}{\partial x} \tag{4}$$

onde q designa a taxa de transferência de calor por unidade de área, K_{fibra} a condutividade térmica e dT/dx o gradiente de temperatura estabelecido na fibra ao longo do seu comprimento. Designando por ρ_{fibra} e c_{fibra} respectivamente a massa volúmica e o calor específico da

fibra, tem-se que a taxa de acumulação de energia interna no interior da fibra é dada por

$$\frac{\partial}{\partial t}\oint_{V}\rho_{fibra}c_{fibra}TdV \qquad (5)$$

Para que haja conservação da energia é necessário que esta taxa de acumulação seja simétrica do fluxo de q através da superfície da fibra, ou seja, que se verifique a condição seguinte:

$$\rho_{fibra}c_{fibra}\frac{\partial T(x,t)}{\partial t}+\frac{\partial q}{\partial x}=0 \qquad (6)$$

Por substituição de (5) em (6) obtém-se a equação de difusão do calor a uma dimensão.

$$\frac{\partial T(x,t)}{\partial t}=\frac{K_{fibra}}{\rho_{fibra}c_{fibra}}\frac{\partial^2 T(x,t)}{\partial x^2} \qquad (7)$$

sendo que a fracção $K_{fibra}/\rho_{fibra}c_{fibra}$ designa a constante de difusão, D. Para a situação em estudo, deve dar-se atenção ao facto de que há fontes de calor em contacto com a fibra e também trocas com o exterior, tornando-se por isso necessário acrescentar no segundo termo da equação 7 uma função que represente essa consequente taxa local de variação de temperatura. Assim sendo, a equação de difusão do calor é modificada para a forma [4]:

$$\frac{\partial T(x,t)}{\partial t}=D\frac{\partial^2 T(x,t)}{\partial x^2}+H(x,t) \qquad (8)$$

Onde H é uma função dada por:

$$H(x,t)=-h\left[T(x,t)-T_{amb}\right] \qquad (9)$$

A grandeza representada por h é designada de coeficiente de transferência de calor por convecção e radiação.

A obtenção de uma expressão analítica para o coeficiente de transferência de calor por convecção é de elevada complexidade uma vez que este depende da geometria do sistema, sendo geralmente obtido de forma empírica por recurso a análise dimensional e resultados experimentais [3]. Para a situação em estudo, tem-se que a potência transferida da fibra para o ar na sua envolvente por convecção livre é dada por [5]:

$$\frac{\partial q}{\partial t}=\left[T(x,t)-T_{amb}\right]\pi L_{fibra}K_{ar}Nu \qquad (10)$$

em que L_{fibra} e K_{ar} designam respectivamente o comprimento do segmento de fibra e a condutividade térmica do ar e Nu o número de Nusselt. O coeficiente de transferência de calor por convecção fica definido pela equação 11.

$$h_{con}=\frac{\pi L_{barra}K_{ar}Nu}{c_{barra}m} \qquad (11)$$

A derivação do coeficiente de transferência de calor para a radiação baseia-se na equação de Stefan-Boltzmann para um corpo cinzento, cujo enunciado matemático é dado pela expressão seguinte:

$$P_{rad}=\sigma Ae\left[T^4(x,t)-T_{amb}^4\right] \qquad (12)$$

sendo A a superfície radiativa, e a emissividade cujo valor está compreendido entre 0 e 1 e σ a constante de Stefan-Boltzmann. O termo entre parêntesis no segundo membro da equação 12 pode ser simplificado fazendo a sua expansão numa série de Taylor de primeira ordem de acordo com:

$$\left[T^4(x,t)-T_{amb}^4\right]\approx\sigma Ae\left.\frac{\partial T^4(x,t)}{\partial T}\right|_{Tamb}\left[T(x,t)-T_{amb}\right] \qquad (13)$$

A potência irradiada também se pode expressar de acordo com a equação:

$$P_{rad}=mc_{fibra}\frac{\partial T(x,t)}{\partial t} \qquad (14)$$

onde m representa a massa da barra.

A equação de Stefan-Boltzmann pode reescrita tendo em atenção as equações 12-14.

$$\frac{\partial T(x,t)}{dt}=\frac{4\sigma AeT_{amb}^3}{c_{barra}m}\left[T(x,t)-T_{amb}\right] \qquad (15)$$

A substituição da equação 15 em 7 permite encontrar o coeficiente de transferência de calor por radiação.

$$h_{rad}=\frac{4\sigma AeT_{amb}^3}{c_{barra}m} \qquad (16)$$

Tomando um valor de emissividade igual a 0,5, A como a superfície lateral de um cilindro com 3 cm de comprimento e 125 μm de diâmetro e o número de Nusselt dado pela aproximação de Kramer igual a 0,39 [5], tem-se que o coeficiente h vale 0.0197s^{-1}.

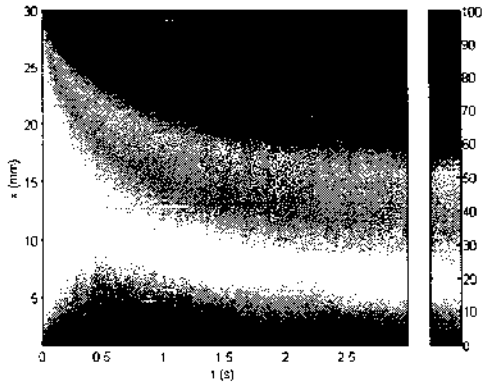


Figura 2- Evolução temporal da temperatura ao longo da barra para T_1=0ºC, T_2=100ºC e T_{amb}=25ºC

A equação de difusão do calor foi então implementada numericamente usando o método das diferenças finitas nas abordagens explícita, implícita e de Crank-Nicholson.

A figura 3 mostra a evolução da temperatura ao longo da fibra com o tempo, sendo possível a distinção entre uma fase transiente e outra estacionária alcançada por volta de 2,5 s. No estado estacionário os valores de temperatura com a posição ajustaram-se segundo uma função linear.

B. Modelização da dispersão

É possível ajustar de forma dinâmica o valor da dispersão na rede de Bragg introduzindo perturbações que se reflictam na largura espectral da rede. A aplicação de gradientes de temperatura em redes de Bragg gravadas em fibra óptica origina alterações espectrais devido à sensibilidade térmica da sílica. As alterações produzidas pelo efeito da temperatura, podem ser obtidas pela diferenciação da equação 1 em torno de uma temperatura de referência.

$$\lambda_B(x) = 2n_{eff}\Lambda(x)\{1+\alpha_n\Delta T(x)\}\{1+\alpha_\Lambda\Delta T(x)\} \quad (17)$$

onde $\Delta T(x)$ é a diferença de temperatura num dado ponto da CFBG e a temperatura de referência considerada igual a 25ºC (a temperatura à qual a CFBG foi gravada). α_n é o coeficiente termo-óptico considerado igual 8.3×10^{-6} ºC^{-1} para Sílica dopada com Germânio, sendo α_Λ o coeficiente de dilatação térmica, igual a 0.55×10^{-6} ºC^{-1}.

A expansão da equação 17 representa um polinómio de grau 3 que pode ser aproximado por outro do segundo grau, desprezando o produto $\alpha_\Lambda\alpha_n$ que é aproximadamente igual a 10^{-12} ºC^{-2}. Assim sendo, a posição local na CFBG onde uma dada componente espectral é reflectida é dada pela seguinte equação [6]:

$$x(\lambda) = \frac{-a_1 \pm \sqrt{a_1^2 - 4a_2a_0(\lambda)}}{2a_2} \quad (18)$$

onde os coeficientes a_1, a_2 and a_3 são dados por:

$$\begin{aligned} a_2 &= C_\Lambda(\alpha_n+\alpha_\Lambda)\left(\frac{T_2-T_1}{L}\right) \\ a_1 &= \Lambda_0(\alpha_n+\alpha_\Lambda)\left(\frac{T_2-T_1}{L}\right) + C_\Lambda + C_\Lambda(\alpha_n+\alpha_\Lambda)(T_1-T_{ref}) \\ a_0(\lambda) &= \Lambda_0\left[1+(\alpha_n+\alpha_\Lambda)(T_1-T_{ref})\right] - \frac{\lambda}{2n_{eff}} \end{aligned} \quad (19)$$

Tomando como referência a componente espectral reflectida em x=0, tem-se que o atraso de grupo se obtém a partir da equação 18 através de:

$$\tau(\lambda) = \frac{2n_{eff}}{c}x(\lambda) \quad (20)$$

Atendendo a que a dispersão se define como a taxa de variação do atraso de grupo com o comprimento de onda, é possível obter analiticamente a dispersão e a sua derivada fazendo a primeira e a segunda derivada da equação da equação 20.

IV. RESULTADOS E DISCUSSÃO

Calculou-se a temperatura local na CFBG através da metodologia apresentada na secção anterior, realçando contudo que as suas extremidades não se encontram efectivamente em contacto térmico com os TEC mas sim afastadas de 3,25 mm. Tomou-se o espectro da reflectividade da CFBG à temperatura constante de 25ºC, de forma a obter a sua largura espectral (4.06 nm), tendo-se ajustado o valor do índice de refracção efectivo de modo a que todas as componentes do espectro sejam reflectidas numa posição na rede compreendida entre 0 e L. A implementação das equações 18, 20 e suas sucessivas derivadas permite a determinação da dispersão para uma dado par de temperaturas (T_1, T_2) nas suas extremidades.

Os resultados do modelo foram validados com dados experimentais obtidos através da caracterização do dispositivo compensador aqui apresentado. Os espectros da reflectividade e atraso de grupo foram obtidos pelo método de desvio de fase [7], sendo que a dispersão foi tomada como o declive de uma recta tangente ao gráfico do atraso de grupo. A figura 3 ilustra o espectro do atraso de grupo para valores de temperatura nos extremos iguais a (0, 60)ºC, (60, 0)ºC e (25, 25)ºC, sendo que a temperatura ambiente é constante e igual a 25ºC.

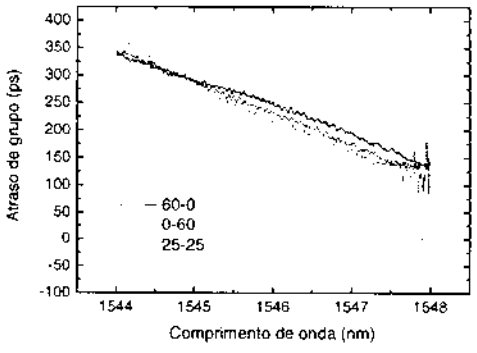


Figura 3. Espectros experimentais do atraso de grupo para temperaturas aos extremos respectivamente iguais (0, 60)ºC, (60, 0)ºC e (25, 25)ºC.

Os resultados obtidos para a dispersão usando os dois procedimentos encontram-se sumariados na tabela I.

Tabela I
Comparação entre os valores obtidos teoricamente e experimentalmente para a dispersão na CFBG.

T_1, T_2 (°C)	$D_{experimental}$ (ps/nm)	D_{modelo} (ps/nm)
0, 60	-49.81	-48.01
60, 0	-66.51	-65.99
25.25	-56.92	-56.00

V. CONCLUSÕES

Mostrou-se que a utilização de gradientes de temperatura em dispositivos compensadores baseados em redes de Bragg de período variável providencia dispersão dinamicamente reconfigurável. Para tal, modelizou-se a distribuição de temperatura ao longo de uma CFBG cujos extremos se encontram a temperaturas definidas e cuja superfície lateral está em contacto térmico com o exterior, tendo-se observado a existência de um estado transiente seguido de outro estacionário, sendo que neste a distribuição de temperatura é uma função linear da posição. A dispersão de uma CFBG nestas condições foi então modelada, e posteriormente validada com resultados experimentais.

REFERÊNCIAS

[1] A. Othonos, K. Kalli, "applications of Bragg Grattings in Communications," *Fiber Bragg gratings: fundamentals and applications in telecommunications and sensing*, Artech House, 1999.

[2] P. S. André, A. N. Pinto, Chromatic Dispersion Fluctuations in Optical Fibers Due to Temperature and Its Effects in High-Speed Optical Communication Systems, Optics Communications, 2004.

[3] M. Kaufman, Priciples of thermodynamics, Marcel Dekker, New York, 2002

[4] R. H. Landau; M. J. Páez, Computacional physics problem solving with computers, Wiley, New York, 1997

[5] F. Noppenberger, M. Still, H. Venzke, "Influence of humidity on hot wire measurements", Meas Sci Tecnology U. K., vol 7, pp1517-] modulation formats", submitted to Journal of high speed networks.

[6] Berta Neto, Luís M. Sá, R. N. Nogueira, João Lemos Pinto, M. J.N. Lima , F. Da Rocha , A. L. J. Teixeira, Paulo André , "Thermodynamic model of tunable chromatic dispersion compensator using chirped fiber Bragg grating" in Conftele 2005

[7] D. Derickson, Fiber Optic Test and Measurement, Prentice Hall PTR, New Jersey, 1998.