Faculdade de Engenharia da Universidade do Porto

Arquitectura de sistema de vigilância integrada

Daniel Filipe Martins Durães

Versão Provisória

Dissertação realizada no âmbito do
Mestrado Integrado em Engenharia Electrotécnica e de Computadores
Major Telecomunicações

Orientador: Prof. Luís Corte-Real
Co-orientador: Eng Luís F. Teixeira

Junho de 2008

Resumo

A segurança nos tempos de hoje é um capítulo importante na agenda dos governos de vários países desenvolvidos a nível mundial. Aperfeiçoar os sistemas de segurança, torná-los mais eficazes e inteligentes é cada vez mais um progresso necessário.

A presente dissertação, "Arquitectura de sistema de vigilância integrada", foi realizada no Instituto de Engenharia e Sistemas e Computadores do Porto (INESC Porto).

Com este trabalho pretende-se projectar e especificar uma arquitectura de um sistema de vigilância inteligente, capaz de analisar informação proveniente de diversas fontes. Um outro objectivo desenvolvido nesta dissertação prende-se com a criação de um protótipo funcional de um sistema de vigilância inteligente.

Este texto apresenta-se elaborado de forma a detalhar como foram atingidos estes objectivos. Primeiramente, fez-se uma análise da evolução dos sistemas inteligentes de vigilância até ao seu estado da arte, bem como as suas técnicas mais utilizadas.

Para desenvolver a presente dissertação, efectuou-se uma análise comparativa de alguns sistemas de vigilância inteligentes mais recentes.

Ao elaborar a arquitectura pretendida, são levantados vários tipos de requisitos e apresentados alguns diagramas usando modelos UML. Através destes requisitos é possível fazer uma análise do problema e encontrar uma solução para a arquitectura de um sistema de vigilância baseado em múltiplas fontes de informação.

A implementação de um protótipo foi baseada no *software* livre de videovigilância, ZoneMinder, no qual foram realizados testes e apresentados resultados do desenvolvimento do protótipo funcional.

Abstract

Nowadays, the security is a very important chapter for governments around the world. Improving the security systems and making them more efficient and intelligent is a necessary progress.

This dissertation, with the title "Architecture of integrated surveillance system", was done at Instituto de Engenharia e Sistemas e Computadores do Porto (INESC Porto).

This dissertation proposes to specify a surveillance system design and architecture based on multiple sources of information. Another aim is the prototype creation of the proposed architecture, based on ZoneMinder.

This text presents how the objectives were achieved. Firstly, a review on the evolution of intelligent surveillance systems and most used techniques was done. A comparative analysis of some surveillance systems was also done.

To produce the desired architecture, various types of requirements were specified using UML diagrams. Through these requirements, it is possible to analyze the problem and construct a solution to the architecture of a multi-source surveillance system.

The implementation of a prototype was based on the free *software* for video surveillance, the ZoneMinder. Finally, tests were performed to validate the functional prototype.

Agradecimentos

Primeiramente, gostaria de agradecer ao Prof. Luís António Pereira de Meneses Corte-Real, orientador da FEUP, e ao Eng. Luís Filipe Pinto de Almeida Teixeira, responsável pelo acompanhamento, o apoio dado ao longo da dissertação.

Por último, agradeço a minha família e amigos, o apoio e disponibilidade que demonstraram durante a realização desta dissertação.

Índice

Índice de figuras

Capítulo 1

1. Introdução

Este capítulo contém uma descrição da dissertação, nomeadamente o seu enquadramento, motivação, objectivos e sua estrutura.

1.1. Enquadramento

A eficácia dos sistemas de vigilância é uma das preocupações mais recentes dos países para evitar desastres, ou até mesmo atentados terroristas. O recente interesse pela vigilância pública, militar e comercial impulsionou o desenvolvimento da videovigilância inteligente.

Sendo esta uma preocupação actual, encontrar soluções eficientes e cada vez mais sofisticadas é uma necessidade dos nossos tempos, face ao elevado crescimento do número de câmaras de vigilância bem como outro tipo de sensores.

Os sistemas vigilância inteligente são sistemas capazes de monitorar em tempo real informação proveniente de diversas fontes. Estes sistemas permitem uma interpretação automática das imagens vídeo, podendo detectar objectos suspeitos ou comportamentos pouco adequados. Para atingir estes fins, são usados algoritmos específicos tais como: detecção e reconhecimento de objectos, *tracking* e análise comportamental.

Os sistemas mais recentes de vigilância, ainda não respondem a todas as necessidades. Nomeadamente no que diz respeito ao processamento e integração de diversas fontes de informação. Torna-se assim relevante um estudo e o de uma arquitectura que suporte os requisitos mais exigentes.

1.2. Motivação

Sistemas baseados em câmaras de segurança, têm-se preferido, porém o número de câmaras instaladas superou a capacidade humana de analisar em tempo-real as imagens provenientes das mesmas. Assim, surge a necessidade de um sistema inteligente capaz de explorar os dados de diversas interfaces, tais como sensores, microfones, imagens e outros, de forma a tomar automaticamente uma decisão ou activar um alarme. Para isso recorre-se à elaboração ou aperfeiçoamento de algoritmos, de maneira a tornar os sistemas de segurança inteligentes mais seguros e autónomos. Com o desenvolvimento e aperfeiçoamento destas técnicas, torna-se inevitável o papel cada vez mais passivo do operador humano num sistema de vigilância inteligente.

Para este tipos de sistemas de vigilância ainda existe um longo caminho de desenvolvimento pesquisa e implementação, pois ainda é necessário trabalho de investigação ao nível da arquitectura, das telecomunicações como no aperfeiçoamento de algoritmos.

Assim, esta dissertação procura ser mais um contributo para o desenvolvimento deste tipo de sistemas, impulsionando uma nova arquitectura que toma decisões através de múltiplas entradas de informação.

As dificuldades e a complexidade deste tipo de sistemas também representam um factor de motivação, dado que este trabalho visa estar ao serviço das pessoas, contribuindo para que no futuro possa ser útil para a sua vigilância.

1.3. Objectivos

Pode-se então destacar os principais objectivos desta dissertação através dos seguintes tópicos:

- Especificar um sistema inteligente de vigilância que integre informação de diferentes fontes, permitindo ainda navegar através do histórico de eventos detectados;
- Implementar um protótipo funcional, baseado no *software* ZoneMinder adicionando um algoritmo de *segmentação* e outro de *tracking*.

Para atingir os objectivos propostos será necessário aprofundar alguns temas, nomeadamente:

- A evolução dos sistemas de vigilância inteligente;
- Análise do *software* ZoneMinder.

1.4. Contribuições

As principais contribuições desta dissertação são:

- Estudo das técnicas e das arquitecturas de vigilância inteligentes usadas actualmente;
- Análise comparativa de sistemas de vigilância existentes no mercado;

- Levantamento dos requisitos necessários de um sistema de vigilância inteligente;
- Estudo da arquitectura funcional do ZoneMinder;
- No decorrer da dissertação foi escrito um artigo com o título "Building Modular surveillance systems based on multiple sources of information" para o SIGMAP 2008 (International Conference on Signal Processing and Multimedia Applications), no qual foi aceite. (Anexo A)

1.5. Organização

A estrutura da dissertação pode ser dividida em 6 partes:

Evolução dos sistemas de vigilância inteligentes – Neste capítulo é apresentada a evolução dos sistemas de vigilância.

Técnicas usadas nos sistemas de vigilância - Demonstra as técnicas utilizadas nos sistemas de vigilância inteligentes.

Análise comparativa de sistemas de vigilância mais recentes – Neste capítulo efectua-se uma análise crítica e comparativa de sistemas de vigilância mais recentes.

Arquitectura de um sistema inteligente de vigilância que integre informação de diferentes fontes – Levantamento dos requisitos funcionais e não-funcionais e a especificação da arquitectura de um sistema inteligente de vigilância.

Implementação – Neste capítulo é feita uma análise detalhada do funcionamento do ZoneMinder, bem como os algoritmos de *segmentação* e *tracking*. É ainda pormenorizado a integração destes algoritmos com o ZoneMinder e demonstrado os resultados práticos.

Conclusão – Neste capítulo é feita uma análise crítica dos resultados obtidos e das contribuições resultantes da investigação efectuada ao longo da dissertação. É também apresentada uma discussão sobre possível trabalho futuro, bem como dificuldades sentidas e meios de superação das mesmas.

Capítulo 2

2. Sistemas de vigilância inteligentes

2.1. Evolução dos sistemas de vigilância

Ao logo dos tempos, sentiu-se a necessidade de melhorar a técnicas de vigilância, para que estas usufruíssem do progresso da tecnologia e das telecomunicações, permitindo melhores resultados. Assim, a evolução dos sistemas de vigilância inteligentes pode-se dividir em três gerações. A primeira geração utiliza técnicas analógicas, a segunda faz uma transição entre as técnicas analógicas e digitais, convergindo actualmente na terceira geração que desfruta do tratamento digital de vídeo e outras interfaces, adicionando ainda o poder de decisão e alerta ao sistema.

2.1.1. Primeira geração

A primeira geração de sistema de vigilância começou com os sistemas CCTV (Closed Circuit Television) analógicos. Este sistema consiste na conexão de um número considerável de câmaras a um monitor através de um switch. Tipicamente o monitor encontra-se numa sala de controlo que é supervisionada por uma ou mais pessoas. Actualmente a maioria destes sistemas usam técnicas analógicas para o transporte de imagem até à sala de controlo. No entanto as câmaras convencionais usadas pelos sistemas CCTV usam CCD (charge-coupled device), que consiste num sensor para a gravação de imagens formado por um circuito integrado. Sendo assim este sistema captura uma imagem em formato digital e de seguida é convertida num sinal de vídeo analógico. Através de cabos coaxiais, as câmaras são ligadas a uma matriz de CCTV onde podem ser manipuladas ou gravadas.

Esta conversão digital para analógico causa uma considerável degradação na imagem. Adicionalmente o sinal analógico é mais susceptível ao ruído.

Actualmente o desenvolvimento dos sistemas de primeira geração centram-se na pesquisa analógico versus digital, na gravação digital e na compressão de vídeo.

2.1.2. Segunda geração

A segunda geração é fruto do desenvolvimento e melhoramento dos sistemas CCTV para sistemas semi-automáticos. A segunda geração é baseada na criação de algoritmos robustos na área de análise de comportamentos, como por exemplo, o sistema de videovigilância em tempo real, capaz de combinar técnicas como detecção de objectos ou grupos de pessoas, analisando os seus comportamentos.

A segunda geração acrescenta uma maior eficiência aos sistemas CCTV. No entanto também se levantam problemas, sendo esta uma geração baseada em algoritmos de detecção, *tracking* e análise de comportamentos humanos conduz ao problema de como tornar estes algoritmos robustos.

2.1.3. Terceira geração

A terceira geração tem como objectivo a concepção de sistemas de vigilância automatizados capazes de cobrir grandes áreas, através de sistemas distribuídos e heterogéneos. Trata-se de um sistema multi-sensor que possui numerosas entradas de dados, tendo ainda autonomia para gerar alarmes automáticos.

Ao contrário dos sistemas de primeira e segunda geração, a terceira geração tem como objectivo desenvolver tecnologias de compreensão automática de vídeo, permitindo a um único operador humano controlar comportamentos humanos nas mais complexas áreas civis.

O processamento de imagem acoplado a técnicas de processamento de sinal, permite o transporte sobre uma rede de dados usando dispositivos embutidos, que oferece uma maior escalabilidade e robustez nos sistemas distribuídos.

Neste tipo de sistemas alguns problemas exigem mais estudo, nomeadamente no que respeita à integração da informação que se obtém a partir dos diversos sensores: estabelecimento da correspondência do sinal no espaço e no tempo; coordenação e distribuição de tarefas; comunicação de vídeo, design e metodologia a adoptar para estes tipos de sistemas de forma a corresponder a uma eficiência cada vez melhor.

Recentemente, a rápida evolução dos sistemas sem fios (por exemplo WiFi, WiMax e bluetooth) e a proliferação da internet tornam possível a transmissão de vídeo sobre IP favorecem a oportunidade para o desenvolvimento em larga escala de Distributed Video Surveillance (DVS). Actualmente muitas empresas comerciais já fornecem soluções de videovigilância baseadas em IP, aproveitando o facto da acessibilidade à internet poder ser feita hoje em dia a partir de qualquer lugar.

Actualmente as pesquisas nesta geração centram-se na fusão de dados, no duelo inteligência centrada versos distribuída e nas técnicas de vigilância que envolvem múltiplas câmaras. A tendência no futuro será melhorar o processamento de imagem para que um sistema seja capaz de realizar tarefas cada vez mais complexas. Aumentar a robustez dos algoritmos de detecção,

reconhecimento de objectos, *tracking* e reconhecimento de comportamentos humanos, são outras áreas de investigação, tal como criar novas soluções para a comunicação de sistemas de vigilância distribuídos.

2.2. Técnicas usadas nos sistemas de vigilância

Segundo Valera and Velastin (2005), apresenta as técnicas básicas usadas na videovigilância. Genericamente os sistemas de vigilância integrados apresentam os seguintes módulos:

- Detecção de objectos
- Reconhecimento de objectos
- Tracking
- Análise comportamental
- Base de dados

Fig. 1- Blocos de baixo nível necessários para um sistema de vigilância distribuído. (Valera and Velastin, 2005)

2.2.1. Detecção de objectos

As duas técnicas mais usadas para detecção de objectos são a diferença temporal e a subtracção de fundo. A primeira consiste na subtracção de duas frames consecutivas e a segunda técnica consiste em subtrair o fundo da imagem de forma a identificar o objecto.

2.2.1.1. Diferença temporal

A diferença temporal consiste em calcular a diferença entre a "frame" corrente e a n-ésimo "frame" anterior (André, 2004).. Quanto maior o n mais espesso será o contorno. Considerando que uma imagem pode ser representada da seguinte forma.

$$I = f(x,y,t) \quad (1)$$

onde I é um valor que representa a intensidade luminosa no ponto de coordenadas (x,y) no instante t, tem-se que, como mostrado em (2).

$$\Delta I = |f(x,y,t2) - f(x,y,t1)| \quad (2)$$

onde ΔI é o modulo da diferença entre a imagem no instante t2 (imagem actual) e a imagem no instante t1 (imagem anterior).

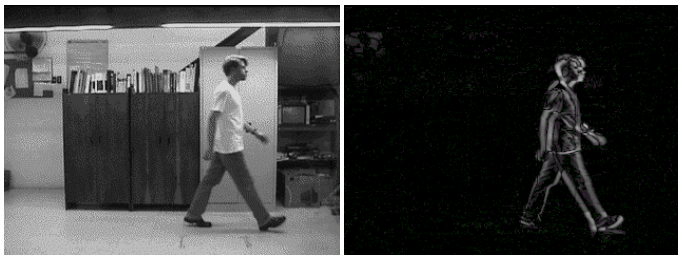


Fig. 2- À esquerda frame actual, à direita diferença entre a frame actual e o anterior. (André, 2004)

Num ambiente perfeitamente controlado, com pouca variação, a diferença entre duas frames consecutivas tem um valor igual ou muito próximo de zero. Eestas regiões correspondem aos locais onde existe pouca ou nenhuma variação da intensidade luminosa, ou seja, o fundo de uma frame para a outra permaneceu estático. Da mesma forma quando os valores da variação da intensidade luminosa são diferentes de zero, as respectivas regiões correspondem a objectos em movimento. A partir daqui, tendo já sido detectado um objecto, seria necessário, na fase seguinte identificá-lo.

A diferença temporal consiste numa abordagem simples. Este método mostra-se rápido mas pouco robusto a alterações de iluminação. Trata-se de um método com má performance para ambientes dinâmicos.

Contudo, esta técnica apresenta alguns problemas, no que respeita à detecção, esta apenas consegue detectar objectos em movimento e revela-se uma técnica pobre em extrair todos os pixéis relevantes dos objectos a detectar.

2.2.1.2. Subtracção de fundo

A subtracção de fundo consiste em seleccionar uma simples frame que não contenha objectos em movimento ou qualquer outro tipo de objecto que não pertença ao fundo. Esta frame irá representar a cena de fundo. De seguida é calculada a diferença entre cada pixel da frame actual e a frame de fundo. Se esta diferença for maior que um limiar, este pixel é considerado como sendo pertencente ao objecto a detectar, caso contrário considera-se que este pixel pertence ao fundo.

2.2.1.2.1. Estimação do fundo

Uma solução simples para obter o modelo do fundo seria filmar o cenário sem os objectos ou estabelecer uma cor específica para o fundo. Contudo ambas as soluções são difíceis de concretizar em cenários não controlados como por exemplo numa via pública. Outra solução mais realista seria utilizar um subconjunto de frames e através da média ou da mediana obter o fundo. A média é simples e eficiente mas é influenciada por valores discrepantes. A mediana consiste numa estatística mais robusta, no entanto implica maiores recursos computacionais e de armazenamento.

Outra abordagem possível consiste num modelo de fundo que não é baseado num subconjunto de frames, inicialmente é atribuído ao modelo de fundo a primeira frame da sequência, e a cada nova frame da sequência, o modelo de fundo é actualizado. Desta forma, o modelo de fundo vai-se adaptando às mudanças de iluminação e do ambiente. Ainda na construção de ambientes de fundo existem diversas técnicas, sendo as mais comuns o modelo gaussiano simples ou o modelo gaussiano múltiplo, os quais serão descritos na seguinte secção.

2.2.1.2.2. Método Gaussiano Simples ou Modelo Pfinder

A ideia do Pfinder (Hall et al., 2005) , consiste em realizar uma etapa de ajustamento. Durante esta etapa é necessário o ambiente estar limpo de objectos em movimento. Durante o ajustamento, são calculados para cada pixel x a sua média $\mu_0(x)$ e sua matriz de co-variância $K_0(x)$. No período de treino é feita a actualização estatística de cada pixel a cada frame processado. A média para o pixel x no i-ésimo frame da sequência é actualizada da seguinte forma:

$$\mu_i(x) = (1-\alpha)\mu_{(i-1)} + \alpha\, f_i(x) \quad (3)$$

onde $\alpha \in (0,1)$ é a taxa de aprendizagem e $f_i(x)$ é o valor do pixel x na frame i.

A matriz da co-variância para cada pixel x é actualizada pela seguinte expressão:

$$K_i(x) = (1-\alpha)K_{i-1}(x) + \alpha v_i(x)v_i(x)^T \quad (4)$$

onde $v_i(x) = f_i(x) - \mu_i(x)$ e α é a taxa de aprendizagem utilizada.

Para definir se um determinado pixel x pertence a um objecto ou a um fundo estabeleceu-se um limiar. A pertença ou não desse pixel ao limiar (5), define se este pertence ao objecto ou ao fundo, respectivamente.

$$l(x) = -\frac{1}{2}v_i(x)^T K_i(x)^{-1} v_i(x) - \frac{1}{2}\ln|K_i| - \frac{3}{2}\ln(2\pi) \quad (5)$$

O Pfinder é muito limitado e com resultados pouco satisfatórios. Os seus principais problemas são: não suportar mudanças na iluminação, não permitir mais de um único objecto em movimento e esperar mudanças lentas na geometria do fundo. Como vantagens apresenta uma abordagem simples e uma boa recuperação da imagem de fundo.

2.2.1.2.3. Método Gaussiano Múltiplo

O método de misturas de Gaussianos de (Stauffer and W. Grimson, 200) é usado para modelar cenários complexos e variáveis com o tempo. Quando temos uma única fonte de luz que influencia a intensidade de um pixel, o modelo gaussiano é suficiente para modelar o fundo. Contudo, não é isto que acontece, a intensidade do pixel resulta na reflexão de múltiplas fontes de luz que variam durante a sequência de imagens. Assim o modelo gaussiano de misturas pode absorver por exemplo o ruído de um pixel numa sequência de imagens, ou um simples galho de uma árvore que balança com o vento. Este método torna a captação do fundo mais robusta e

dinâmica, visto que, numa cena realista o ambiente é dinâmico. Como desvantagem deste sistema é o custo computacional envolvido.

2.2.1.2.4. Algoritmos

Para proceder à detecção de objectos através da subtracção de fundo, pode-se recorrer à (1) Subtracção Básica de Fundo Estimado pela Média, (2) Subtracção Básica de Fundo Estimado pela Mediana, (3) Subtracção de Fundo pela Média Adaptativa, (4) subtracção de Fundo pela Mediana de um Intervalo. (Lara,2006)

A primeira (1) técnica estima o fundo a partir da média, este algoritmo é rápido, porem não é muito robusto, podendo-se atingir um bom resultado com algumas sequências embora para outras o resultado não seja tão bom. O uso da média para estimação do fundo não é o ideal, visto que esta é susceptível a valores discrepantes.

A segunda (2) técnica consiste num algoritmo mais robusto em relação a valores discrepantes na estimação do fundo, visto que esta se baseia na mediana. No entanto, torna-se insuficiente para situações como por exemplo alterações na iluminação.

A terceira (3) técnica consiste num algoritmo mais robusto em relação às alterações de iluminação, bem como ao estimar o cenário de fundo. Um dos problemas está na demora em adaptar novas alterações de cenário, ao fundo.

A quarta (4) técnica tem como característica a adaptação às alterações de cenário e de iluminação. Consiste no cálculo da mediana de uma sequência de imagens. Mostrando-se um algoritmo bastante robusto.

2.2.2. Reconhecimento de objectos, *tracking*

O *tracking* consiste num sistema cujo objectivo é seguir um ou mais objectos através de frames consecutivas, de forma a determinar a trajectória do seu movimento no cenário ou caracterizar o seu movimento relativamente a outros objectos (Fig. 3). (Foresti, 1999)

As técnicas usadas para reconhecimento de objectos podem ser divididas em duas abordagens principais: os modelos 2-D, com ou sem forma explícita e modelos 3-D.

O modelo 3-D utiliza uma abordagem explícita a priori, que é usada em sistemas de vigilância para detectar pessoas, veículos ou ambos.

A abordagem a priori consiste primeiro no *tracking* de objectos, depois através de um conhecimento a priori de por exemplo, forma, regras de movimento, informações sobre a altura, etc.. Este método revela-se bastante robusto na aplicação a ambientes internos ou externos, bem como em relação a mudanças de iluminação. Existem ainda sistemas capazes de combinar os modelos 2-D e 3D.

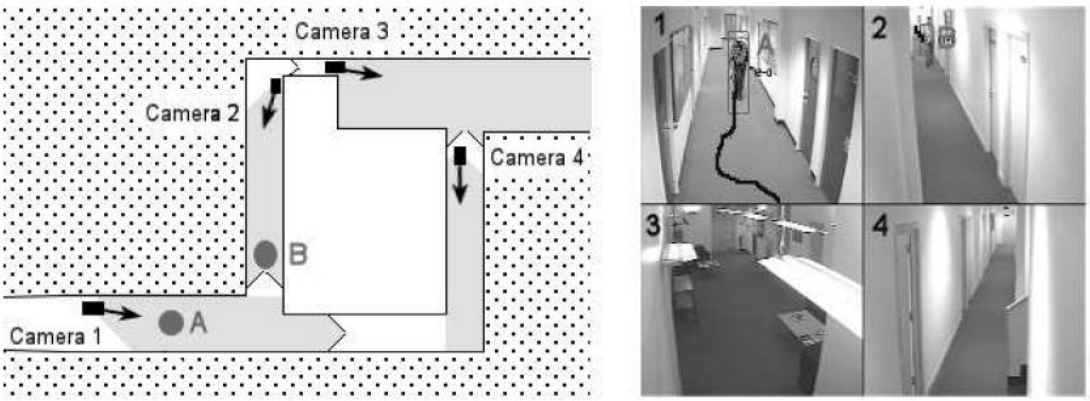


Fig. 3-Exemplo de *Tracking* entre câmaras

2.2.2.1. Processo de captura do movimento humano

O processo de captura do movimento humano pode dividir-se em 4 etapas.

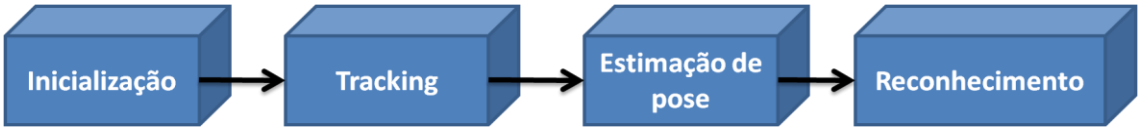


Fig. 4– Blocos constituintes do processo de captura humano

A inicialização consiste na adaptação da câmara ao meio, calibração da câmara, adaptação das características do cenário.

O *tracking* consiste na detecção do indivíduo, seguido da representação da imagem de forma apropriada. Diferentes estratégias (modelos) são usadas para que a pessoa seja seguida ao longo do tempo.

A estimação de pose é um processo cujo objectivo consiste em identificar o corpo ou os membros de uma pessoa. Este processo pode ser executado recorrendo a modelos humanos, através de técnicas baseadas em conhecimento a priori.

O reconhecimento consiste na classificação da postura ou do movimento.

2.2.2.2. Problemas Comuns

Apesar da boa performance e eficácia destes sistemas, existem alguns problemas que actualmente estão a ser estudados com o objectivo de proporcionar um algoritmo mais robusto. Os problemas mais comuns são as mudanças de iluminação, as sombras ou reflexos, a distância do objecto à câmara, rápidas mudanças de direcção ou velocidade, o movimento de pessoas em grupo ou ainda a ocultação temporária parcial ou total.

2.2.3. Análise de comportamentos

Este bloco dos sistemas de videovigilância consiste no reconhecimento e na compreensão de comportamentos dos objectos seguidos. Para fazer uma análise inteligente do cenário e para reconhecer as suas actividades, os sistemas de vigilância necessitam de efectuar uma variedade de tarefas para caracterizar o objecto. Por exemplo, num ambiente urbano este sistema tem que reconhecer pessoas, grupos de pessoas ou veículos.

Nesta fase procede-se à classificação do problema ao longo do tempo, através dos dados fornecidos pelas fases anteriores. Este processo consiste em corresponder uma sequência de modelos pré-compilados e rotulados numa biblioteca ao cenário em análise. Esta biblioteca contém protótipos de acções que necessitam de ser identificados e rotulados pelo sistema de vigilância.

Existem várias abordagens tais como o Dynamic time warping (DTW) (Rath and Manmatha, 2003) ou Hidden Markov models (HMM) (Oates Schmill and Cohen, 2000).

A primeira consiste num algoritmo para medir a semelhança entre duas sequências, sequências estas que podem variar no tempo ou no espaço. O DTW tem sido aplicado para vídeo, áudio e gráficos. É ainda capaz de reconhecer discurso para lidar com as diferentes velocidades de discurso.

Este sistema é bastante robusto para um conjunto de sequências lineares, contudo para sequências não lineares no tempo e no espaço não é tão eficaz como o sistema HMM.

A abordagem HMM é um modelo estatístico, em que o processo a ser modulado pressupõe um processo de Markov com parâmetros desconhecidos. O desafio consiste em determinar esses parâmetros ocultos a partir de parâmetros observáveis. O modelo extraído pode ser usado para futuras análises, tais como, o reconhecimento de padrões.

2.2.4. Base de dados

A base de dados de um sistema de vigilância corresponde ao armazenamento e recuperação de aspectos importantes que foram detectados ao longo destes módulos. Actualmente pouca investigação foi feita relativamente à forma de armazenar e recuperar todas as informações obtidas através do sistema de vigilância. Torna-se um aspecto relevante, especialmente quando neste tipo de sistemas é possível ter na base de dados diferentes tipos e formatos de dados e informação a recuperar.

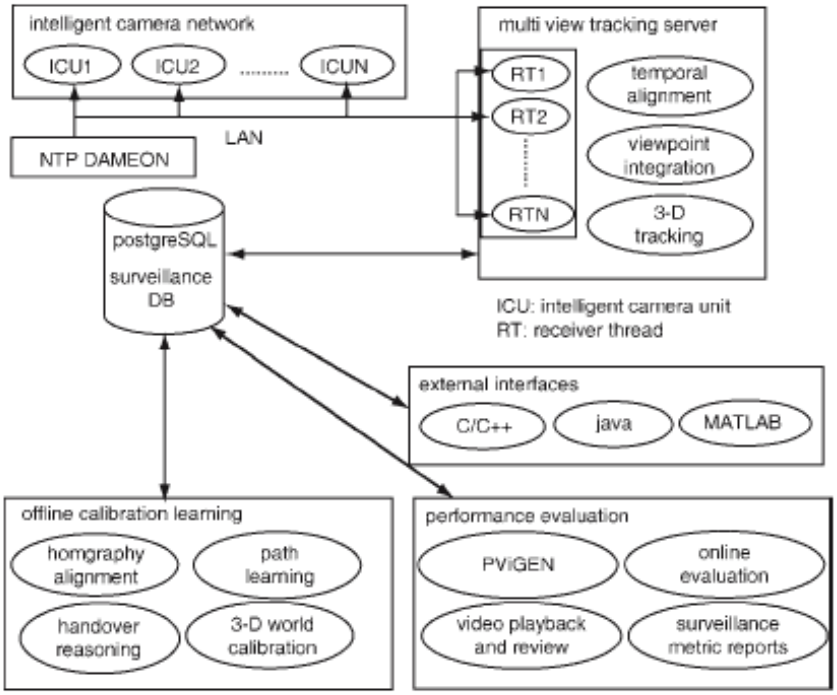


Fig. 5- Arquitectura de um sistema multi-câmara proposto por MaKris e tal. (2004).

EmMaKris e tal. (2004), os autores investigam a definição e a criação de um modelo de base de dados capaz de suportar o armazenamento de dados em diferentes níveis (Fig.5).

As bases de dados para sistemas inteligentes de vigilância ainda se encontram em processo de discussão, não havendo para já uma arquitectura bem definida. Contudo, quaisquer que sejam as vertentes de desenvolvimento, estas devem assentar na eficiência de armazenamento, indexação e recuperação de todas as informações recolhidas pelo sistema.

2.2.5. Resumo

A maior parte dos sistemas de vigilância usados hoje em dia, ainda consistem em sistemas de segunda geração, em que valorizam o processamento de sinal digital e possuem pouca inteligência, tal como foi detalhado no capítulo 2. Porém com o avanço da investigação começam-se a implementar no mercado sistemas de vigilância de terceira geração, sistemas estes capazes de monitorar centenas de câmaras e tratar informação de diferentes interfaces. Também no capítulo 2 são abordados os procedimentos habituais de um sistema de videovigilância de terceira geração, bem como as suas técnicas usadas, contudo estas técnicas revelam ainda uma margem de desenvolvimento e aperfeiçoamento da sua eficácia.

Capítulo 3

3. Análise comparativa de sistemas de vigilância mais recentes

Actualmente existem em investigação e escalonamento sistemas de vigilância inteligentes com diferentes interfaces de entrada (sistemas multi-sensores). Nesta secção serão apresentados e avaliados alguns exemplos de sistemas de vigilância de terceira geração.

3.1. DETER

O DETER (Detection of Events for Threat Evaluation and Recognition) é um exemplo de um sistema comercial aplicado a ambientes externos. (Pavlidis et al., 2001)

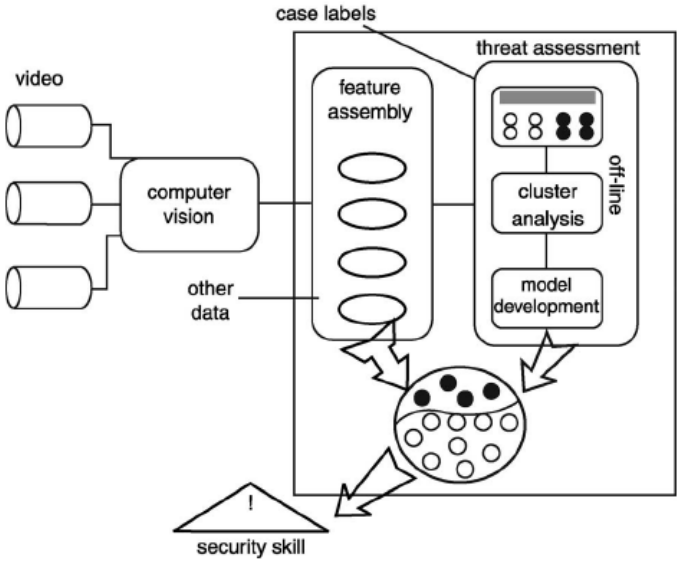


Fig. 6-Arquitectura do sistema DETER. (Pavlidis et al., 2001)

Na Fig. 6 encontra-se esquematizada a arquitectura deste sistema. O sistema consiste em duas tecnologias diferentes, a visão computacional e a avaliação de ameaças. A visão computacional consiste num design óptico e de sistema capaz de segmentar objectos, *tracking* e

a fusão de dados proveniente de múltiplas câmaras. A avaliação de ameaças consiste na aquisição/treino que é feito off-line e posteriormente a classificação da ameaça.

3.1.1. Avaliação

O sistema demonstra um bom desempenho na detecção e reconhecimento de objectos, no entanto emprega um número reduzido de câmaras para ter uma boa relação de custo/aplicação. Este reduzido número de câmaras enfraquece a qualidade do desempenho do sistema.

Para a função de reconhecimento, DETER necessita de câmaras com uma elevada resolução para capturar imagens com um elevado nível de detalhe a fim de proceder ao reconhecimento de carros ou de pessoas.

Após testes a este sistema concluiu-se também que este produz um baixo número de falsos alarmes.

3.2. ADVISOR

O ADVISOR (Annotated Digital Video for Intelligent Surveillance and Optimized Retrieval) consiste num sistema para efectuar o reconhecimento, elaboração de relatórios e arquivamento do suspeito ou de comportamentos suspeitos, através de videovigilância (Valera and Velastin, 2005).E uma das suas aplicações pode ser nas estações de metro.

Para atingir os seus objectivos, este sistema de vigilância inteligente utiliza a captura de imagens através de CCTV para construir um retrato do comportamento das pessoas em cena.

Este sistema armazena todas as saídas de vídeo de câmaras e, em paralelo com esse armazenamento, são guardadas anotações e eventos considerados relevantes em determinadas sequências. Quando armazenado, o arquivo de vídeo pode ser consultado através de pesquisa de anotações, ou de acordo com horários específicos, podendo obter-se a respectiva sequência de vídeo dessas consultas.

Na Fig.7 podemos ver uma possível arquitectura deste sistema. Como podemos observar, a arquitectura possui uma rede de unidades ADVISOR instaladas em diferentes estações de metro. Cada módulo efectua a detecção e reconhecimento de objectos, "*tracking*", análise de comportamentos e armazena estes dados na sua base de dados.

O ADVISOR não possui um estrutura centralizada (único ponto de interligação das entradas), ou seja cada rede tem o seu próprio CPU que controla cada nó, evitando assim em caso de avaria de um nó prejudicar os restantes nós.

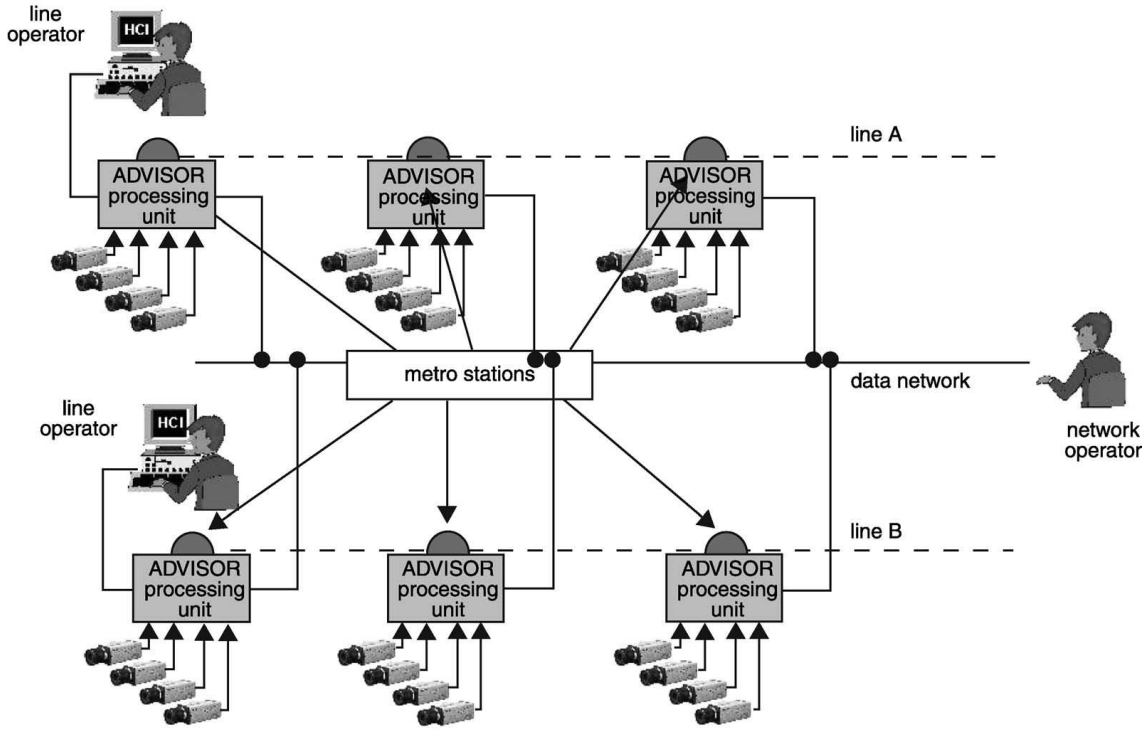


Fig. 7- Arquitectura ADVISOR (Valera and Velastin, 2005).

3.2.1. Avaliação

Tem uma arquitectura inteligente característica dos sistemas de terceira geração, contém também uma arquitectura descentralizada, o que tem vantagens relativamente a falhas em nós. No entanto trata-se de um sistema semi-distribuído, ou seja, apenas utiliza imagens provenientes das câmaras CCTV, não englobando outro tipo de sensores.

3.3. PRISMATICA

A PRISMATICA (Pro-active Integrated Systems for security Management by Technological Institutional and Communication Assistance) foi fundada pela união europeia com o objectivo de tornar os transportes públicos mais atraentes e seguros para os passageiros. Este sistema de vigilância inteligente é aplicado aos transportes públicos. (Velastin e tal., 2005)

Ao contrário de outros sistemas este sistema não é apenas baseado em vídeo, ou seja, consiste num sistema multi-sensor que incorpora vídeo para além de sensores de áudio. (Fig. 8)

Características do PRISMATICA:

- Distribuição de processamento, dada a extensão geográfica dos sensores e do poder computacional.

- Integração dos diferentes tipos de dispositivos num sistema flexível de arquitectura, devido à variedade de fontes de informação que são necessárias para apoiar a tomada de decisões e de apoiar futuras melhorias no desenvolvimento de dispositivos de detecção;
- Convergência da informação numa interface humana para o operador navegar e recuperar dados necessários (base de dados).

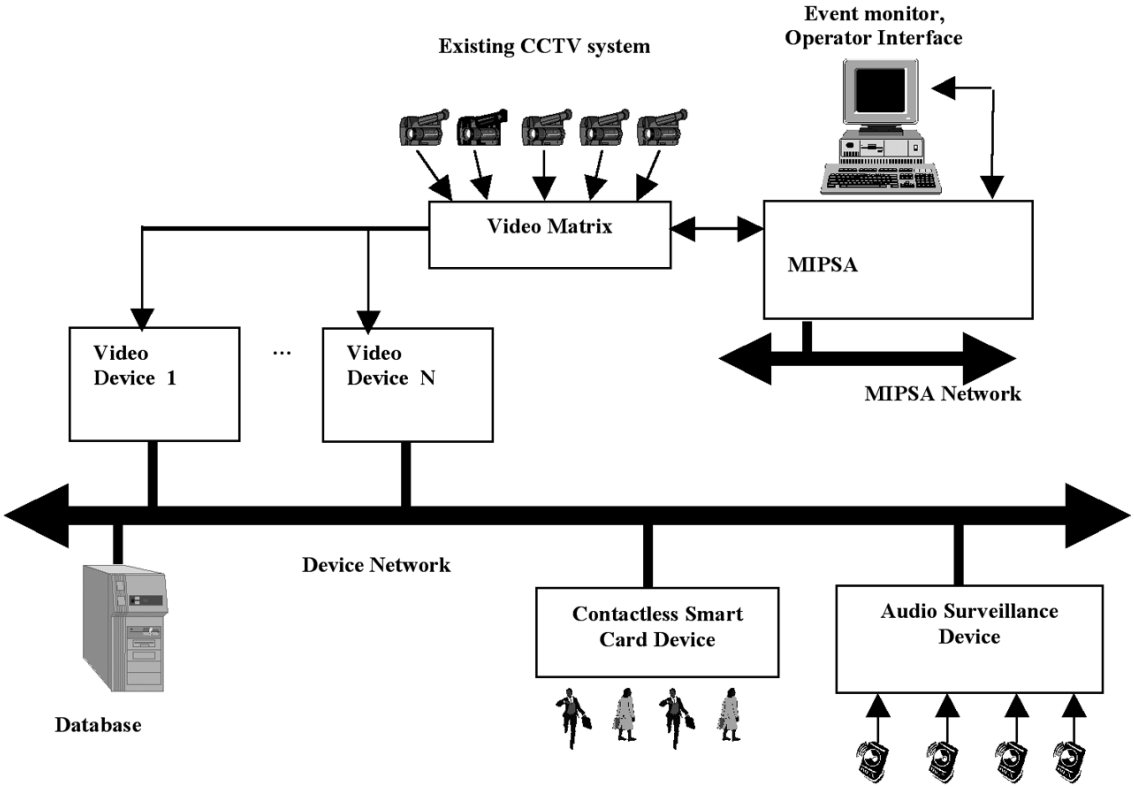


Fig. 8– Componentes do sistema PRISMATICA. (Velastin e tal., 2005)

A conexão entre os sistemas CCTV, "smart cards" e os sensores áudio podem ser feitas através de wireless.

Todas as entradas deste sistema estão ligadas a um processamento central o MIPSA (Modular Integrated Passenger Surveillance Architecture), este bloco faz a fiscalização necessária para coordenar acções e para recolher informações geradas a partir dos dispositivos, de modo a ajudar os processos de decisão. O MIPSA proporciona um único ponto de contacto com o operador e um meio de controlo e comunicação com dispositivos inteligentes.

Uma matriz de vídeo recebe os sinais provenientes do sistema CCTV que são controlados pelo MIPSA. Este sistema também inclui um dispositivo para captar sinais de cartões inteligentes. Ou seja, sempre que um passageiro transporta um destes cartões, através de um duplo clique sobre um botão, serão enviados para a MIPSA todos os dados e informações. Através destas informações a MIPSA pode localizar a "chamada" e gerar um alarme, ou exibir imagens das câmaras.

O sistema de áudio também foi concebido para detectar sons invulgares, sons que podem resultar de catástrofes, discussões, etc.. Da mesma maneira após detectar tais eventos, os dados

são enviados para o MIPSA, que em seguida localiza o evento, gera um alarme e mostra as imagens relevantes.

3.3.1. Avaliação

Sendo este um sistema distribuído especialmente concebido para transportes públicos, a sua performance em cenários realistas é bastante elevada. Apresenta bons resultados no que respeita à detecção de objectos. Possui uma arquitectura modular e escalonável através de *hardware* standardizado.

Contudo o conceito deste sistema assenta numa arquitectura centralizada, em que todo o funcionamento do sistema depende do computador central. Assim sendo, o computador central torna-se num ponto de falha crítica, ou seja, se este computador central avariar, todo o sistema fica inoperacional.

3.4. Knight

Knight é um sistema de vigilância automatizado de múltiplas câmaras, que tem sido desenvolvido pela University of Central Florida's Computer Vision Laboratory. Actualmente está a ser utilizado pelo "Florida Department of Transportation, Orlando Police Department, DARPA Small Business Technology Transfer (STTR) program, and Lockheed Martin Corporation".

Em Javed and Shah (2003) é descrito o Knight como um sistema comercial capaz de detectar e caracterizar objectos em movimento, para isso utiliza o estado da arte das técnicas de visão computacional. Para além de detectar, também apresenta um resumo de frames, bem como uma descrição textual das actividades observadas para um operador humano analisar e tomar uma decisão.

A Fig.9 mostra-nos o diagrama de blocos do Knight, onde se pode ver como a informação flui ao longo do sistema.

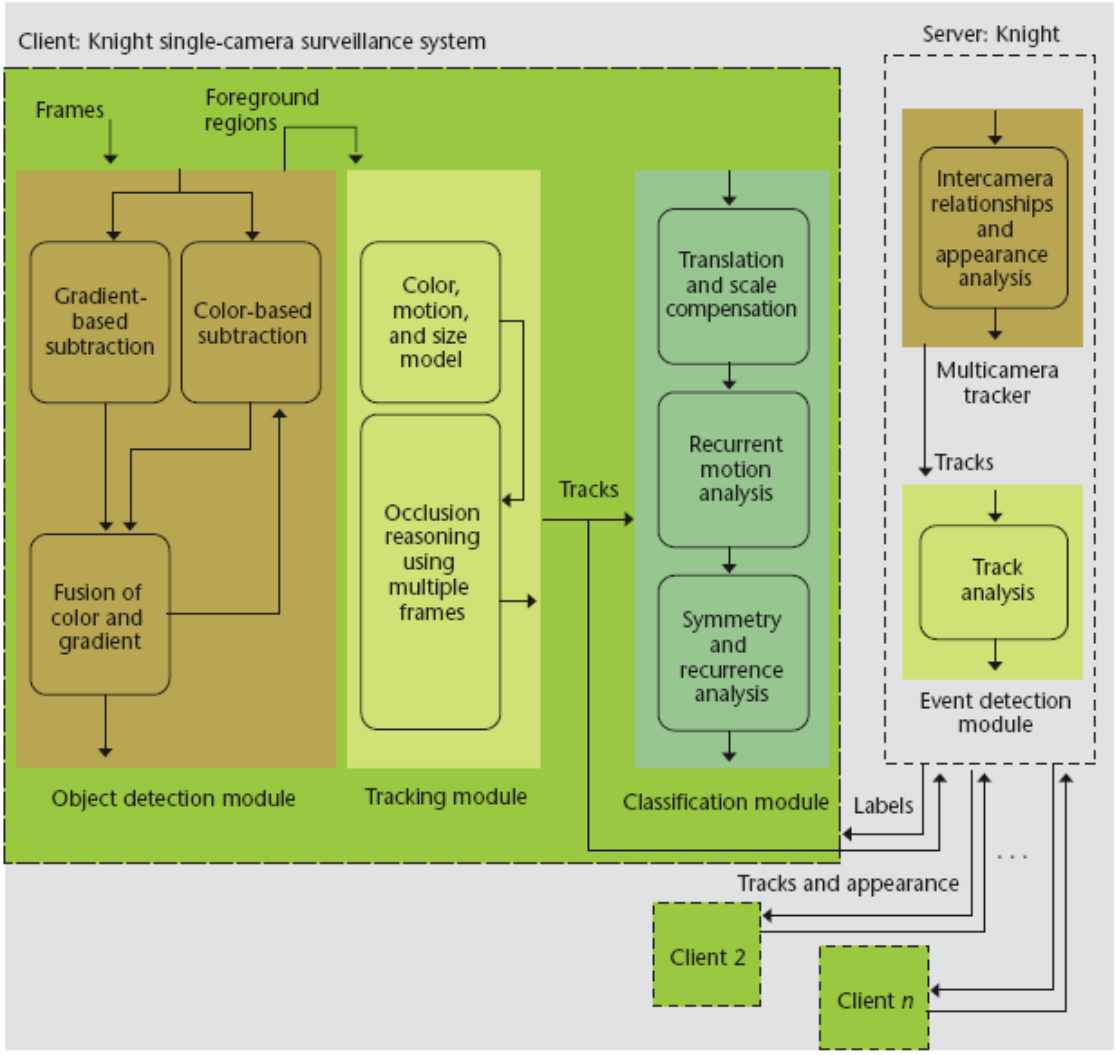


Fig. 9- Diagrama de blocos do Knight. (Javed and Shah 2003)

3.4.1. Avaliação

Este sistema apenas pode funcionar durante o dia, sendo que à noite ou quando a iluminação está abaixo de um certo nível o sistema desliga-se automaticamente. Durante os testes efectuados com este sistema de vigilância foram gerados poucos falsos alarmes, demonstrando um excelente desempenho. Contudo existem falhas, não consegue detectar objectos camuflados e objectos com uma cor similar à cor de fundo. Os testes efectuados em condições meteorológicas desfavoráveis, como por exemplo à chuva, mostraram que condiciona o funcionamento do sistema.

3.5. VIGILANT

Em Greenhill et al.(2002), é apresentado um sistema de vigilância que suporta diversas câmaras.

Este sistema tem como objectivo monitorar pessoas que caminham num parque de estacionamento. Para isso, o sistema faz o *tracking* de pessoas ao longo das câmaras, assim que é detectada uma pessoa é gerada um conjunto de informação acerca da trajectória dessa pessoa.

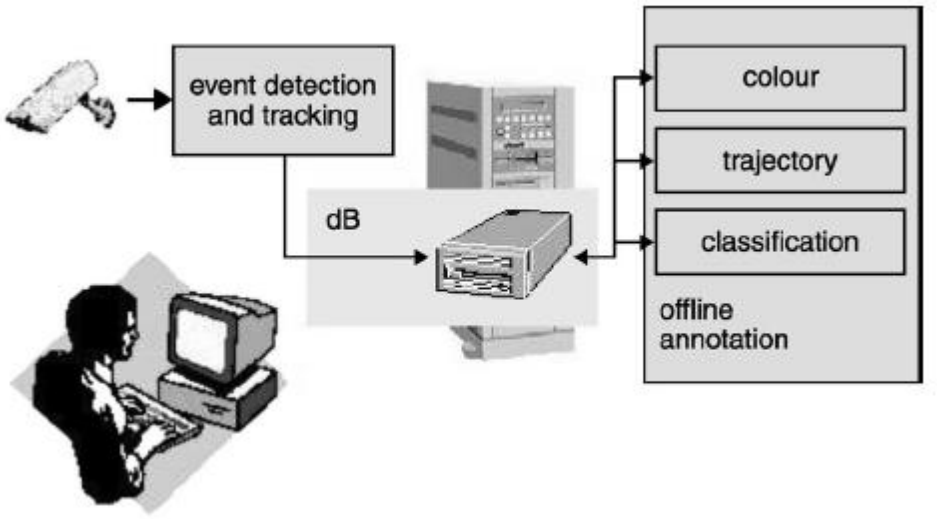


Fig. 10- Arquitectura do sistema VIGILANT. (Greenhill et al.2002)

Como podemos ver na Fig. 10, o sistema VIGILANT, contém a detecção e *tracking* de um objecto em tempo real, e em off-line este anota a cor do objecto, a sua trajectória e a classificação.

3.5.1. Avaliação

Trata-se um sistema um pouco limitado, dado que a sua funcionalidade apenas é referida como detecção e *tracking* de pessoas, contudo para esta tarefa mostra-se eficaz. O VIGILANT sendo um sistema de *tracking* e detecção de pessoas, também deveria conter um módulo de análise comportamental.

3.6. Análise comparativa

Nome do sistema de vigilância	Vantagens	Desvantagens
DETER	-Técnicas de terceira geração. -Boa relação. Custo/Aplicação. -Boa performance.	-Numero reduzido de câmaras. -Sistema semi-distribuído (apenas entradas de vídeo).
ADVISOR	-Técnicas de terceira geração. -Arquitectura descentralizada. -Boa performance.	-Sistema semi-distribuído (apenas entradas de vídeo).
PRISMATICA	-Técnicas de terceira geração. -Arquitectura modular e escalonável. -Sistema multi-sensor. -Boa performance.	-Arquitectura centralizada.
Knight	-Técnicas de terceira geração. -*Tracking* entre câmaras. -Boa performance.	-Funcionamento em condições meteorológicas desfavoráveis. -Apenas sistema multi-câmara
VIGILANT	-Técnicas de terceira geração. -Sistema multi-câmara. -*Tracking* entre câmaras.	-Ausência e módulo comportamental. -Apenas é usado para monitorar pessoas.

Tabela 1. Análise comparativa de sistemas de vigilância de terceira geração.

Capítulo 4

4. Arquitectura de um sistema inteligente baseado em informação de diferentes fontes

Antes de se iniciar o desenvolvimento de qualquer projecto, é necessário um rigoroso levantamento dos requisitos pretendidos com a aplicação. Na primeira parte deste capítulo serão apresentados os requisitos funcionais e não funcionais.

Na segunda parte deste capítulo será especificada uma arquitectura tendo em conta os requisitos definidos. Esta arquitectura será modelada usando diagramas UML.

4.1. Definição de Requisitos

Conforme o objectivo desta dissertação, e após efectuada uma análise aos mais recentes sistemas de vigilância, pretende-se com base neste estudo especificar uma arquitectura para um sistema de vigilância capaz de integrar módulos de análise genéricos, recorrendo a várias fontes de informação, como por exemplo câmaras, sensores, microfones.

A par destes requisitos tenciona-se criar uma arquitectura capaz de escalar facilmente quer ao nível do *software* quer ao nível de *hardware*, contribuindo desta forma para o desenvolvimento de novas técnicas de vigilância.

Primeiramente serão apresentadas as características gerais de uma rede de vigilância e depois será efectuado um levantamento dos requisitos funcionais e não funcionais, bem como os requisitos de *hardware*.

4.1.1. Características gerais de uma rede de vigilância

Actualmente as redes de vigilância assumem um papel importante, pois torna-se necessário monitorizar permanentemente aspectos de vigilância fundamentais, detectar fraudes, etc.

Para cumprir eficazmente essas funções as redes devem, entre outras, conter as seguintes características:

Estrutura Hierárquica - Este tipo de abordagem, pressupõe que esta arquitectura seja vista como uma pirâmide, em que na base existem diversas câmaras e vários tipos de sensores, e no topo o operador humano. Esta visão permite uma abordagem mais simples para especificar uma arquitectura. Os dados obtidos pelas câmaras e sensores irão ser tratados e transformados ao longo da "pirâmide", até chegar ao operador humano que analisará e dará a última resposta.

Capacidade de reportar automaticamente uma anomalia – Sempre que o sistema de vigilância detecte uma situação anómala, o sistema deverá ser capaz de efectuar um relatório e guardar todos os dados da anomalia num histórico para poder ser consultado posteriormente.

Controlo remoto – Durante o funcionamento normal, a rede de vigilância visa minimizar o trabalho dos operadores humanos. Logo, o sistema deverá ser capaz de ser controlado via *software*.

Capacidade de suportar vários algoritmos – À complexidade da videovigilância estão subjacente vários algoritmos capazes de identificar actividades interessantes para a vigilância. Portanto, na prática os sistemas de vigilância devem suportar diversas funções a operar simultaneamente e em cooperação, como por exemplo *tracking*, subtração de fundo, etc.

Potência e largura de banda – As redes de vigilância normalmente não são muito limitadas em termos de potência e de largura de banda. Contudo, quando se fala de grandes redes, como por exemplo, num aeroporto os seus sensores e câmaras tem que ser capazes de trabalhar continuamente em plena capacidade, envolvendo também servidores dedicados.

4.1.2. Requisitos Não Funcionais

A arquitectura a conceber, deve satisfazer não só os requisitos aplicados a sistemas de vigilância, mas também diversos requisitos não funcionais que irão impulsionar o desenvolvimento de *software* para a vigilância. Podem então considerar-se os seguintes:

Escalabilidade – Com este requisito pretende-se que o sistema seja escalável, quer a nível de *hardware* quer a nível de *software*. Assim sendo, um sistema deste tipo tem que estar preparado para aumentar o número de sensores e câmaras de vigilância consoante as necessidades. Também terá que permitir adicionar novos algoritmos de vigilância, bem como permitir a actualização dos algoritmos já implementados.

Fiabilidade – Dado que se trata de um sistema com um grande número de componentes, existe sempre uma elevada probabilidade de haver problemas no *hardware* (câmaras, sensores, etc..). Consequentemente esta arquitectura deverá possuir elevados níveis de fiabilidade.

Evolução – Um sistema de vigilância deve estar sujeito a mudanças de *hardware* e *software*. Um sistema necessita de evoluir para que seja capaz de ao longos dos tempos responder de forma viável às adversidades.

Integração - Este ambicioso requisito, permite ao sistema interagir com sistemas externos, como por exemplo, ser capaz de controlar elevadores ou controlar os acessos a um edifício. Possibilitando assim que este sistema não opere isoladamente como a maioria dos sistemas existentes no mercado.

Segurança – Um sistema de vigilância de elevadas dimensões está sempre sujeito a ataques. O sistema deve-se apresentar robusto às várias tentativas de evasão.

Usabilidade – Todas as interfaces devem ser intuitivas e desenhadas segundo os mesmos padrões de acesso para facilitar a navegação e criar a habituação à aplicação. As interfaces com o operador devem ser simples e nunca devem conduzir a ambiguidades na escolha de opções e a ajuda de contexto deve estar sempre presente. Contudo, trata-se de um sistema que requer ao operador um conhecimento prévio do sistema. O *software* de vigilância é responsável por detectar uma situação anómala e comunicar ao operador humano. No entanto, ao operador humano cabe a decisão final. Todas as acções tomadas por parte do *software* e por parte do operador deverão ser registadas num histórico.

4.1.3. Requisitos funcionais

Nesta secção serão apresentados os requisitos funcionais do sistema de vigilância distribuído. Para uma melhor organização e percepção, optou-se por dividir os requisitos funcionais em blocos. Assim sendo, os requisitos encontram-se distribuídos por diferentes blocos que possuem diferentes funções e que apresentam maior facilidade de escalonamento.

Os blocos constituintes são:

- Processamento de sinal;
- Análise de dados;
- Controlo e funções de inspecção;
- Armazenamento.

4.1.3.1. Processamento de sinal

Este bloco caracteriza-se por estar mais perto do *hardware*, e compreende as seguintes funções:

Detecção de objectos – Tem como objectivo encontrar objectos no cenário a vigiar. Contém algoritmos como subtracção de fundo ou diferença temporal.

Classificação de objectos – Consiste em rotular o objecto consoante o tipo, por exemplo distinguir entre pessoa ou objectos.

Object *tracking* – Seguimento de objectos em cada câmara.

As saídas geradas por este bloco são importantíssimas para o resto do sistema, visto queo desempenho da restante rede de vigilância é em grande parte limitada pelas suas prestações.

Este bloco encontra-se perto da camada física da rede como por exemplo as câmaras de vigilância, recebendo vídeo em bruto das câmaras e originando saídas de dados relevantes.

4.1.3.2. Análise de dados

A análise de dados trata-se de um bloco superior ao bloco de processamento de sinal. Este bloco a partir dos dados recebidos da camada de processamento de sinal aplica métodos para detectar a existência ou não de anomalias.

Contudo, está ainda a associado este nível a gestão de comunicações, visto que este bloco é responsável pela conexão entre o operador e o *hardware*.

Para a análise de dados definiu-se os seguintes requisitos:

Tracking multi-câmara – Consiste na detecção e seguimento do movimento de um objecto ao longo do tempo em diferentes câmaras.

Detecção de comportamento – Consiste na detecção de comportamentos considerados suspeitos.

4.1.3.3. Controlo e funções de inspecção

Este bloco permite a um operador humano examinar uma possível actividade de perigo mais de perto. Possibilita ao operador controlar diversas interfaces, bem como aceder em tempo real a uma determinada câmara.

Os requisitos funcionais para estes blocos são:

Consulta de Histórico – Este requisito possibilita a um operador consultar o histórico de algum alarme detectado ou de alguma situação crítica detectada, podendo assim, saber pormenores do porquê do alarme, bem como visualizar imagens e comportamentos que deram origem a uma situação crítica.

Controlo de câmara – Permite a um operador aceder em tempo real e controlar remotamente uma ou mais câmaras.

Seguimento de trajecto – Permite ao operador seleccionar um objecto (pessoa, veiculo, etc..) e ver qual o movimento dele ao longo da rede de vigilância.

Controlo de sistemas externos – Este requisito engloba a integração e o controlo de sistemas externos ao sistema de vigilância, como por exemplo elevadores, fechaduras das portas, etc..

4.1.3.4. Armazenamento

Este bloco é responsável por armazenar toda a informação importante obtida pelo sistema, permitindo aos investigadores extrair em detalhe toda a informação no caso da ocorrência de algum incidente. Assim, este bloco tem o seguinte requisito:

Arquivamento – Consiste em arquivar todos os dados recolhidos pelo sistema numa base de dados.

4.1.4. Requisitos de *hardware*

Esta arquitectura pressupõe a partilha de dados obtidos através de diferentes interfaces, interfaces estas que podem variar consoante as necessidades ou os sistema de vigilância. Tipicamente todo este *hardware* está ligado a uma rede. A Fig.11 mostra algum do *hardware* utilizado neste tipo de sistemas.

Fig. 11- Requisitos de *hardware*.

O sistema é composto por diferentes componentes de *hardware* que partilham dados, que depois serão analisados e cruzados. Normalmente estes componentes estão ligados a uma *local area network* (LAN).

Deverá existir uma interface gráfica que facilite o trabalho do operador humano. Para a captura de vídeo existem vantagens em utilizar uma *smart network camera* (Fig.12), pois estas são capazes de fazer a aquisição e processamento de dados. Deste modo os dados de vídeo são

enviados de forma comprimida através da rede, visto que estas podem ligar-se directamente com *switches* ou *hubs*.

Fig. 12- Exemplo de uma Smart Camera.

Quanto ao disco rígido este deve satisfazer também alguns requisitos. Tem que ter capacidade suficiente para armazenar 24h de vídeo. Através do formato MPEG-4 a 25fps para 3 dias necessita-se de 40 GB de espaço no disco rígido. Contudo ainda assim torna-se necessário disponibilizar cerca de 70% a 80% de espaço para efectuar uma pesquisa rápida e recuperar sequências já armazenadas.

Outro requisito importante de *hardware* é a largura de banda da rede, pois um operador pode estar a analisar o vídeo em tempo real e o sistema ainda assim continua a ocupar uma certa largura de banda para o seu funcionamento normal. Tipicamente 2Mbit/s de largura de banda é suficiente, contudo, dada a crescente oferta de imagem de alta definição, existem já no mercado de vigilância câmaras de vigilância de alta definição. A largura de banda necessária para transmissão de imagens de alta definição com compressão é cerca de 2,8 Mbit/s. Com estes resultados sugere-se a instalação de uma LAN, que nos dias de hoje tradicionalmente opta-se por uma Gigabit Ethernet ou no pior dos casos 100Mbits.

4.2. Arquitectura

Tal como se tem discutido até aqui, a arquitectura para um sistema desta complexidade assenta numa hierarquia. E para tornar o problema mais simples de tratar, optou-se por dividir o sistema em módulos, onde cada módulo tem a sua tarefa específica para desempenhar e ao mesmo tempo combina com outros módulos de forma a satisfazer os requisitos pretendidos. A junção de todos estes módulos representa o sistema visto como um todo. A Fig.13 mostra o diagrama de componentes.

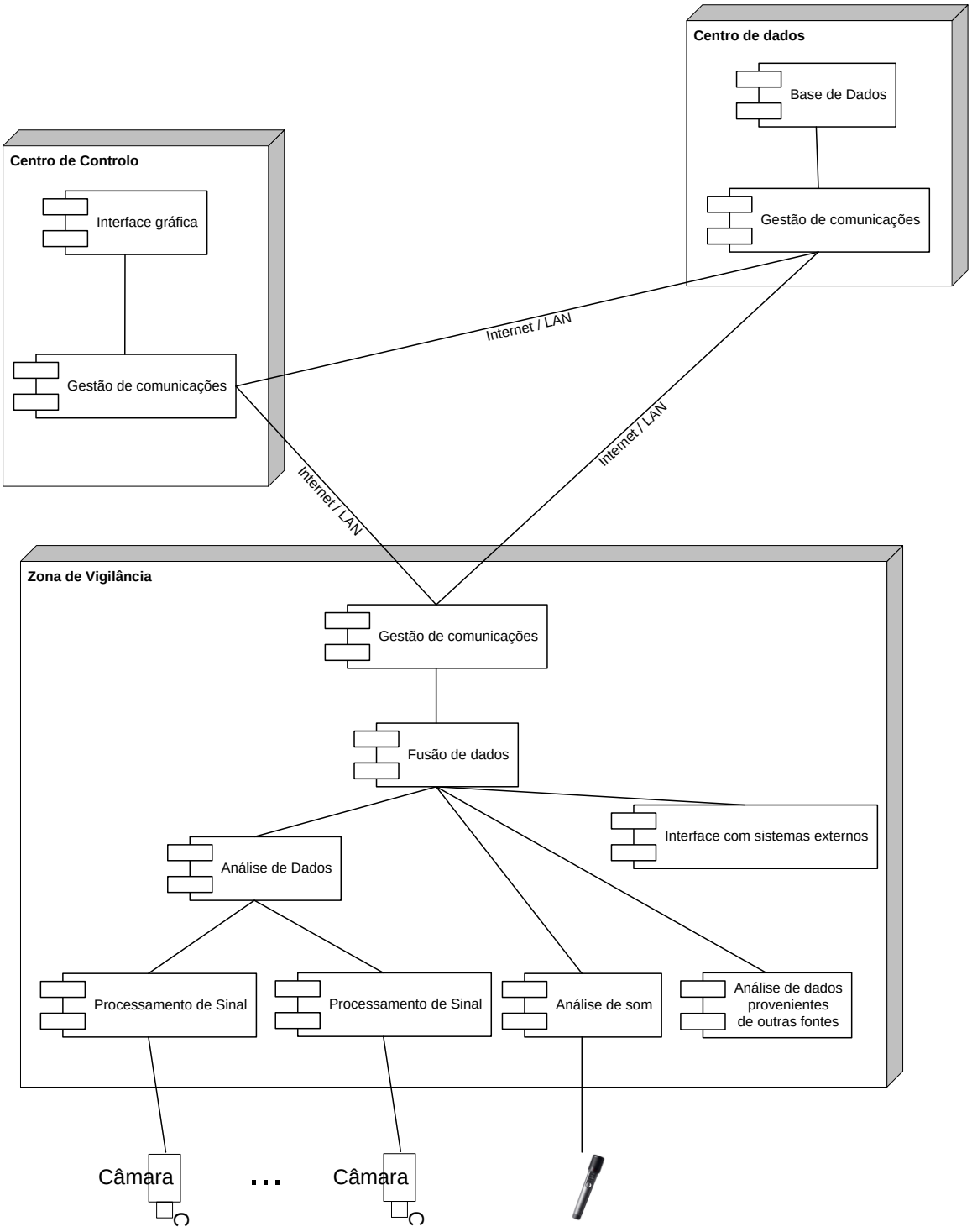


Fig. 13 – Diagrama de Componentes.

No diagrama de componentes identificam-se três nós principais: centro de controlo, centro de dados e zona de vigilância. Estes nós serão explicados detalhadamente de seguida.

4.2.1. Centro de controlo

O centro de controlo é o local onde o operador toma conhecimento dos resultados gerados pelo sistema. Neste local o operador tem ainda disponíveis outras opções de controlo e navegação através de um histórico de acontecimentos.

A interface gráfica é o rosto do sistema, que permite ao utilizador introduzir e receber informação. Para o desenvolvimento deste componente e tendo em conta os requisitos, deverá ser utilizada uma linguagem de alto nível por exemplo Java, Python ou PHP (interface web), integrando-se com os módulos de processamento de sinal, desenvolvidos em C/C++.

O componente de gestão de comunicações, tal como o nome indica, é responsável por fazer a gestão de comunicações entre os diferentes nós do sistema. Este *software* recebe e envia dados, ou seja, dependendo do protocolo utilizado nas comunicações entre os nós, este é responsável por sincronizar, encapsular os dados a enviar, assim como também desencapsular os dados recebidos. (Fig 14)

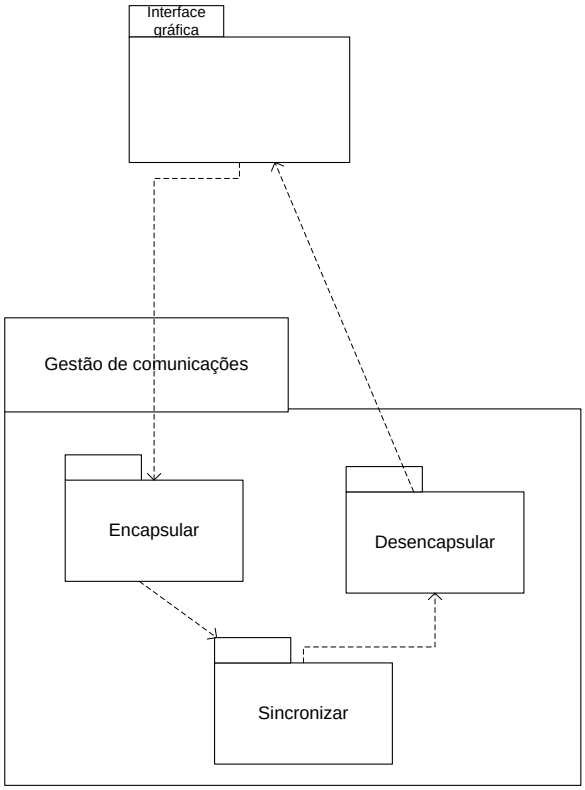


Fig. 14 - Diagrama de pacotes entre a interface gráfica e a gestão de comunicações.

Através da terminologia UML construiu-se um diagrama de casos de utilização (Fig.15) para perceber e modular as opções do operador humano no sistema.

Assim, identificaram-se os seguintes actores:

Operador: Consiste na pessoa responsável por monitorar, operar e interagir com o sistema.

Técnico: É a pessoa responsável por resolver problemas de carácter técnico. Pode ainda actualizar o sistema, bem como fazer a sua manutenção.

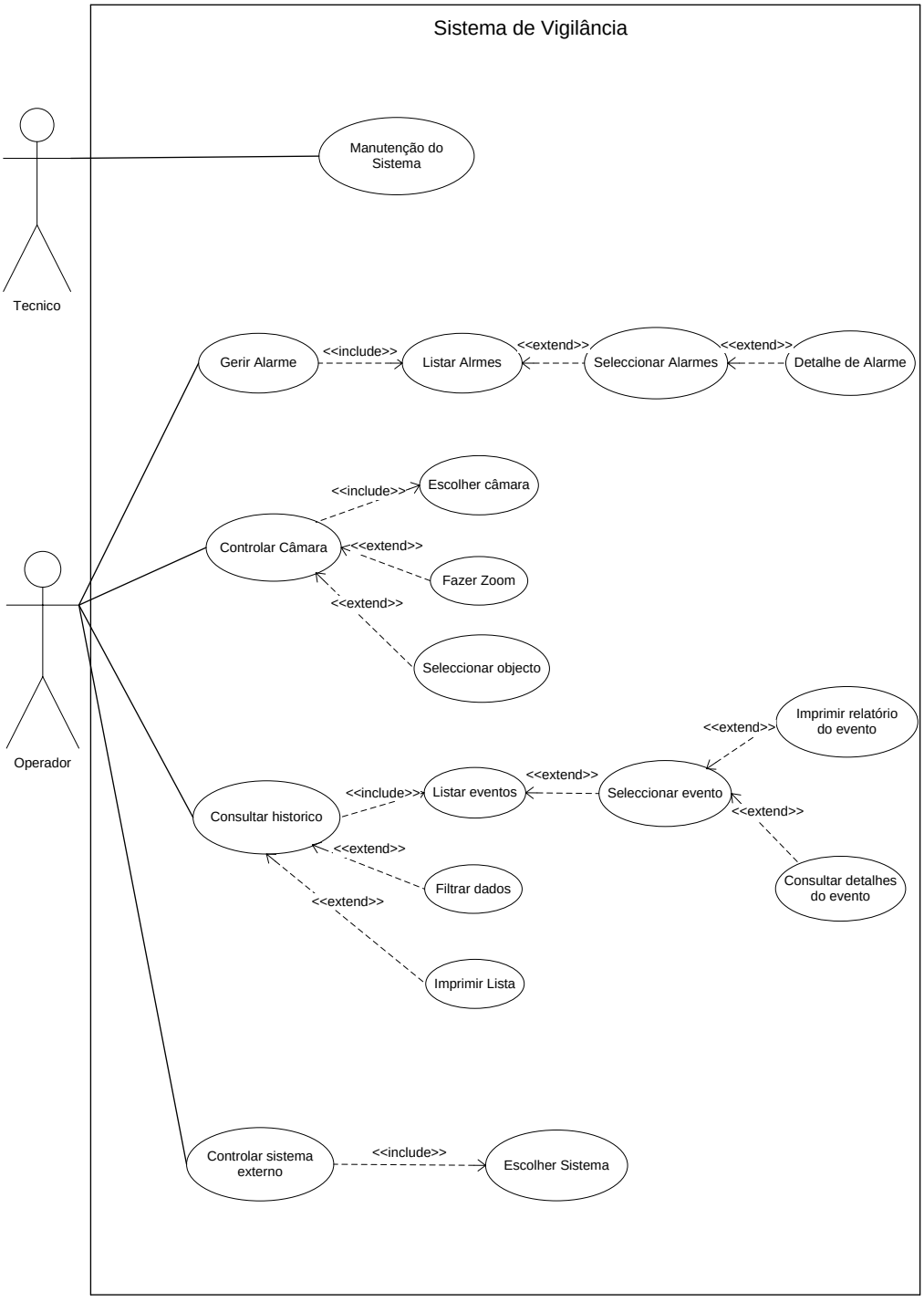


Fig. 15 - Diagrama de Casos de Utilização.

Com este diagrama de casos de utilização, é concretizado observar quais as opções de navegação do operador pelo sistema.

O caso de utilização *gerir alarme*, consiste na opção para quando um ou mais alarmes são lançados pelo sistema. Neste caso o operador pode consultar o, ou os alarmes gerados, e pode ainda aceder de forma detalhada aos acontecimentos que deram origem ao alarme.

O operador também tem acesso a todas as câmaras instaladas no sistema onde, através de uma lista, pode seleccionar uma ou mais câmaras para monitorar em tempo real, podendo ainda fazer zoom. Outra opção do caso de utilização que permite controlar as câmaras é a possibilidade do operador seleccionar um objecto. Se, por exemplo o operador pretender acompanhar o comportamento de alguém suspeito, pode seleccionar o objecto e o sistema fará o acompanhamento e registará todos os movimentos efectuados por esse suspeito.

A opção de consultar um histórico, possibilita ao operador rever alarmes e eventos gerados pelo sistema e que foram armazenados na base de dados. O operador pode filtrar a lista de eventos de forma a obter o resultado mais rapidamente. Seleccionando um determinado evento tem a possibilidade de aceder a uma descrição completa do evento.

Ainda na interface gráfica, é possível ao operador controlar sistemas externos, como por exemplo fechaduras das portas, elevadores, iluminação, e outros. Este requisito permite que o sistema de vigilância interaja com sistemas externos podendo consequentemente controlá-los e tirar benefícios destes sistemas externos.

4.2.2. Centro de dados

O centro de dados consiste numa base de dados onde são armazenados todos os dados que podem ser relevantes para o sistema e para futuras investigações.

Este nó é composto pelo *software* de gestão de comunicações e pela a base de dados que pode ser por exemplo MySQL ou PostgreSQL.

A vantagem de o centro de dados estar representado como um nó é a de que permite a sua independência relativamente ao centro de controlo. Para além disso a base de dados é escalável, podendo ser expandida para conter *n* nós.

O *software* de gestão de comunicações já foi detalhado anteriormente, visto que o seu funcionamento neste nó é semelhante.

Os sistemas distribuídos de vigilância acumulam grandes quantidades de dados em pequenos períodos de tempo. Tipicamente requerem 5 a 10 gigabytes de espaço para um período de 24h. A maioria do espaço ocupado é devido ao vídeo.

4.2.2.1. Base de dados hierárquica

A solução apontada para a base de dados é baseada em Black et al.(2004)**,**. Este pressupõe a criação de uma base de dados com 4 camadas dispostas hierarquicamente que suportam *queries* de alto e de baixo nível.

Este sistema suporta vários tipos de *queries* consoante os resultados que se desejam obter, ou seja, suporta reconhecimento de eventos, sumário de eventos e a monitorização de eventos.

A base de dados do sistema de vigilância está estruturada em quatro camadas de abstracção (Fig.16): *image framelet layer*, *object motion layer*, *semantic description layer* e *meta data layer*. Estas camadas suportam hierarquicamente perguntas enquanto informação é capturada em tempo real. Ao mais baixo nível armazena movimento detectado nas câmaras, enquanto que numa camada a nível superior analisa e responde a complexas *queries*.

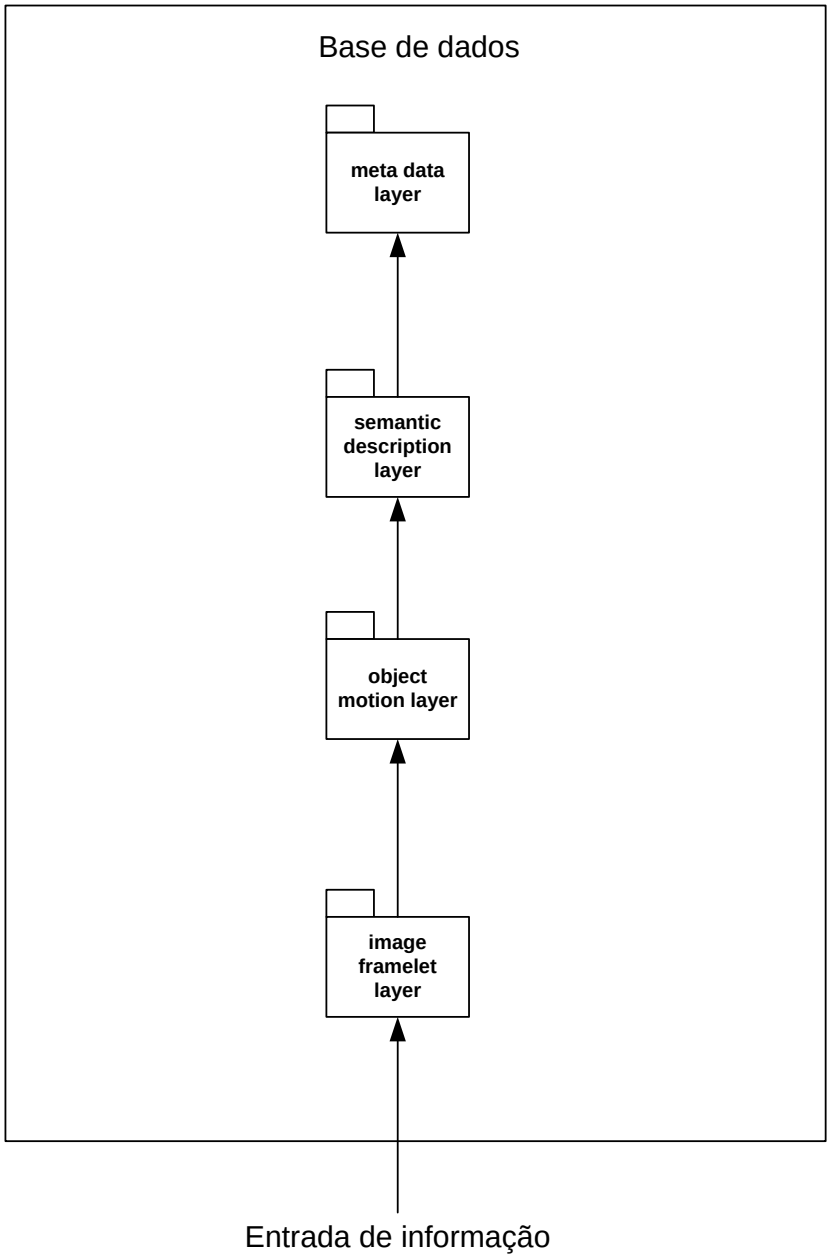


Fig. 16 - Diagrama de pacotes da base de dados.

4.2.2.1.1. Image framelet layer

Esta é a camada de baixo nível que é responsável pelo armazenamento das imagens relevantes. Essencialmente, esta camada guarda a informação proveniente dos resultados da subtracção de fundo ou diferença temporal, que foram transmitidos via TCP/IP em formato MPG-4.

Na Fig.17, podem observar-se alguns objectos armazenados nesta camada. As imagens mostram o histórico dos movimentos ocorridos no campo de vista da câmara.

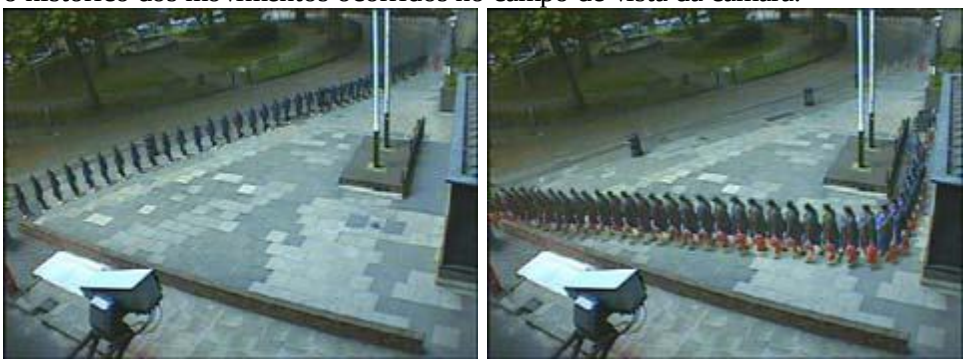


Fig. 17 - Exemplo de objectos armazenados na camada image farmelet. (Black et al.,2004)

4.2.2.1.2. Object motion layer

A camada de object motion encontra-se no segundo nível, e tem como objectivo armazenar os dados resultantes do *tracking* de cada câmara e conjugar com o *tracking* de outras câmaras, obtendo assim a trajectória de um objecto ao longo de várias câmaras.

4.2.2.1.3. Semantic description layer

Esta camada define a região do campo de visão de cada câmara. A informação que chega a esta camada é proveniente do *tracking* da camada de object motion.

A Fig.18 expressa o trabalho desta camada, na qual se identifica a rota de cada objecto, bem como a zona de entrada e de saída. As elipses a preto são correspondentes à zona de entrada, e as brancas correspondem à zona de saída.

Fig. 18 - Caminhos efectuados por objectos. (Black et al.,2004)

4.2.2.1.4. Meta data layer

A base de dados representada na forma de multi-camadas, permite que o vídeo capturado seja armazenado usando uma representação abstracta. É possível gerar dados acerca de dados obtidos noutras camadas, para posterior consulta.

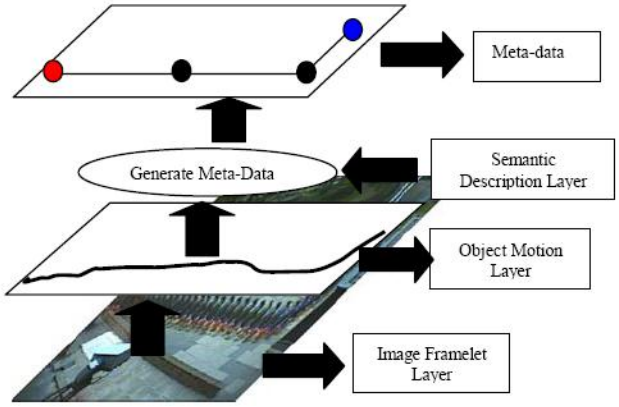


Fig. 19 – Gerar a camada de meta data. (Black et al.,2004)

A Fig.19 mostra a forma como são gerados os dados nesta camada desde a entrada até à camada de meta data. Inicialmente as informações de vídeo e a trajectória dos objectos são armazenadas na camada de *image framelet* e *object motion*. Depois disso o histórico gerado pela camada de *object motion* é armazenada numa linguagem de alto nível através da camada de *semantic description.*

A *meta data layer*, contém a informação de cada objecto detectado, incluindo o ponto de entrada, o ponto de saída o tempo de actividade e as trajectórias detectadas.

4.2.3. Zona de Vigilância

A zona de vigilância corresponde ao nó que faz a interface física com o *hardware*. Pode haver mais do que uma zona de vigilância, podemdo mesmo estar separadas por alguns quilómetros de distância.

Tal como cada um dos outros nós, a zona de vigilância também tem o componente de gestão de comunicações que é responsável por fazer a comunicação nos dois sentidos.

4.2.3.1. Interface com sistemas externos

O componente explicitado neste nó tem como objectivo ligar-se e controlar sistemas externos ao sistema de vigilância. Assim, sempre que um operador por exemplo, através da

interface gráfica, acenda as luzes de uma determinada zona, este componente receberá essa informação, processará e executará.

4.2.3.2. Análise de áudio

Este componente é responsável por comunicar com dispositivos de áudio.

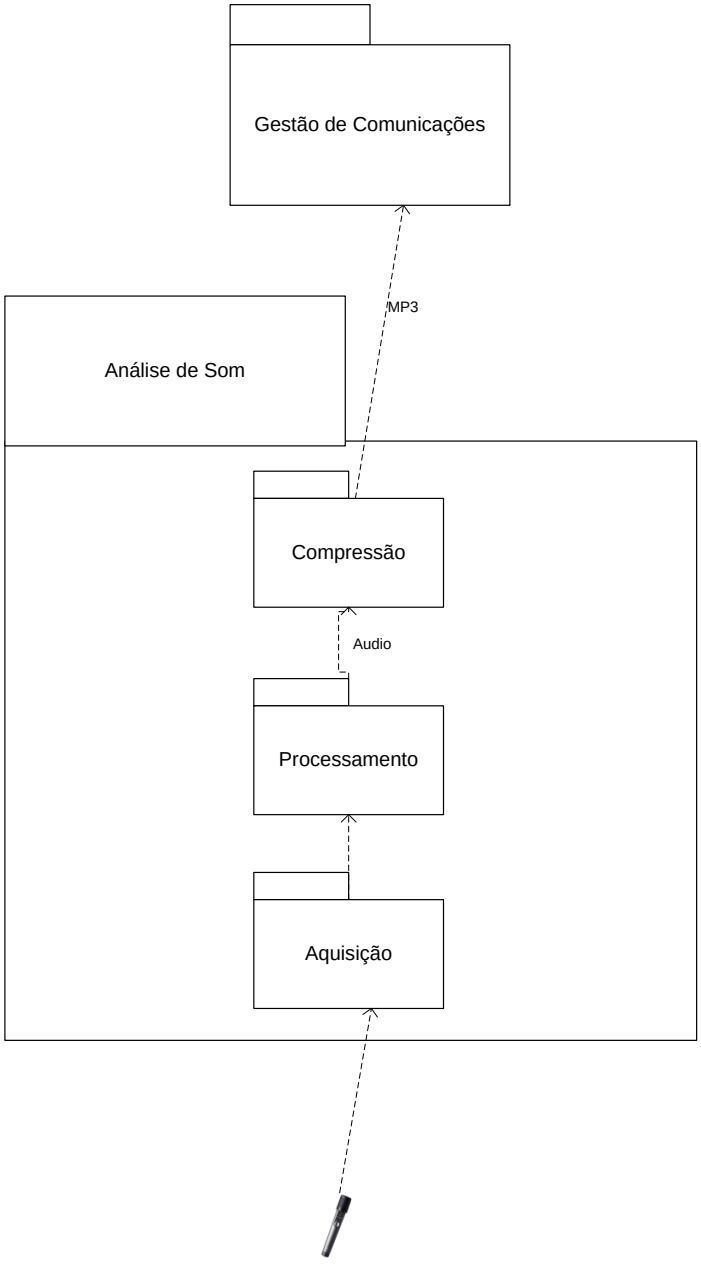


Fig. 20 - Diagrama de pacotes para áudio.

O diagrama de pacotes da Fig.20, ilustra o que se passa no interior do módulo de análise de som. Após aquisição do sinal sonoro, segue-se a fase de processamento e depois uma fase de compressão de dados.

O bloco de processamento consiste num *software* desenvolvido em C++, que é capaz de detectar situações críticas através do som capturado na zona de vigilância. Por exemplo, podem ser detectados gritos e reconhecidas explosões. Sempre que um destes casos críticos é detectado, o bloco de análise de som envia para o centro de controlo um alarme. Posteriormente o operador poderá consultar a base de dados para ouvir o motivo do alarme e aceder a mais detalhes.

O bloco de compressão comprime o sinal de áudio num formato comprimido, por exemplo mp3, de forma a poupar largura de banda quando este é enviado este para a base de dados.

O bloco de análise de som envia dois tipos de resultados, um evento que é gerado quando existe uma situação crítica e um ficheiro áudio encapsulado para armazenar na base de dados.

4.2.3.3. Análise de dados provenientes de outras fontes

Como se trata de um sistema multi-sensor, para além do "tradicional" vídeo, esta especificação deixa um espaço para diferentes tipos de sensores que se queiram utilizar. Este bloco no fundo deixa um espaço para que se alarguem as fontes de dados. Por exemplo, se para o sistema é relevante ter sensores capazes de detectar fumo, este componente importará os dados provenientes deste sensor, e caso seja necessário alertará o operador, enviará uma mensagem ao centro de controlo e outra ao centro de dados para registar o acontecimento.

4.2.3.4. Processamento de sinal

O módulo de processamento de sinal, corresponde a um componente que no seu interior comporta algumas técnicas básicas usadas em sistemas de videovigilância. Consiste num módulo de baixo nível, que é responsável por receber a informação proveniente das câmaras de filmar. A Fig.21 apresenta o conteúdo deste módulo.

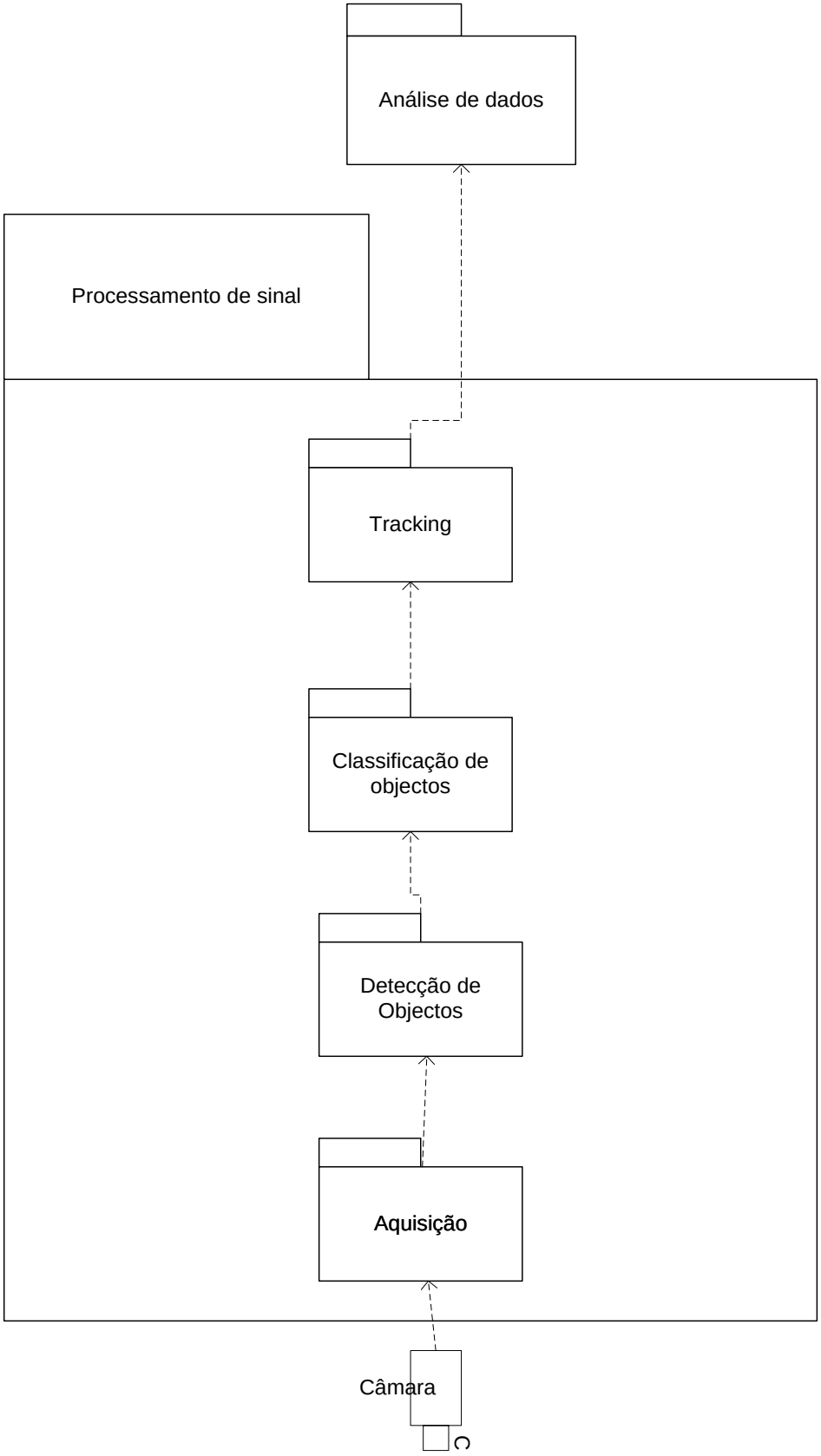


Fig. 21 - Diagrama de pacotes do módulo de processamento de sinal.

Pode-se observar o fluxo de dados ao longo do módulo de processamento de sinal. Assim, primeiramente dá-se a aquisição de dados provenientes das câmaras de filmar, em seguida, as frames capturadas seguem uma trajectória típica dos sistemas de vigilância inteligentes.

Após a aquisição das frames, estas passarão pelo sub-módulo de detecção de objectos. Este sub-modulo faz uma primeira filtragem do cenário, descartando situações em que o cenário está vazio ou que não haja movimento de objectos. Para fazer a detecção de objectos, podem ser utilizados métodos referidos no capítulo 2. Depois de atribuir um destes métodos, podem ainda aplicar-se algumas técnicas de pós-processamento para reduzir o ruído.

Após a detecção de objectos, segue-se o momento de classificar estes objectos, este módulo permite distinguir objectos e classificá-los, por exemplo como uma pessoa ou um veículo.

Ainda no interior do processamento de sinal, mas a um nível mais elevado, é incluído o sub-módulo de *tracking*. Este consiste num algoritmo de *tracking*, que analisa e regista a trajectória de um determinado objecto ao longo das frames capturadas pela câmara.

4.2.3.5. Análise de dados

O módulo de análise de dados, contém algoritmos de vigilância automática de mais alto nível, isto é, este módulo depende das saídas geradas pelo módulo de processamento de sinal.

Na Fig.22, ilustra as funções desempenhadas pelo pacote análise de dados.

Fig. 22 - Diagrama de pacotes da Análise de Dados.

De acordo com o esquema, o módulo de análise de dados recebe dados tratados por parte da camada de processamento de sinal, dados estes que serão agora tratados a um nível superior. Assim, no interior deste módulo temos algoritmos de *tracking multi-câmara* e de *análise de comportamentos*.

O *tracking multi-câmara* é responsável por receber os dados resultantes do *tracking* individual de várias câmaras e cruzar esses dados, de forma a obter a trajectória global dos objectos na zona de vigilância.

A análise de comportamentos engloba algoritmos cuja a função é reconhecer e perceber actividades e comportamentos dos objectos resultantes do *tracking*. Caso seja detectado algum comportamento considerado suspeito, este será comunicado ao operador.

4.2.3.6. Fusão de Dados

O objectivo deste componente é cruzar os dados adquiridos por todo o *hardware* numa dada zona de vigilância.

Este *software* procura perceber se os acontecimentos detectados nas diferentes fontes de informação estão relacionados. Por outro lado, quando surge uma situação crítica é necessário guardar todos os dados na base de dados. Assim sendo, sempre que por exemplo uma câmara detectar algum acontecimento, é enviado para a base de dados não só a informação obtida pela câmara isoladamente, mas também anexar toda a informação recolhida pelos outros sensores presentes no mesmo local. Ao anexarmos esta informação de forma organizada, torna-se possível através da inspecção de um alarme perceber não só o que a câmara detectou, como analisar o áudio anexado no momento da captura das imagens.

O facto de este componente receber dados do componente responsável pela interface com sistemas externos, permite ao próprio sistema de vigilância tomar algumas acções sem intervenção do operador. Ou seja, num dado momento em que uma câmara reconhece algum comportamento estranho, a imagem recolhida pela câmara pode-se tornar mais evidente se a iluminação da zona estiver ligada. Assim, o componente de fusão de dados abre portas a um sistema de vigilância que começa a tomar pequenas decisões que antes apenas eram possíveis através da ordem do operador humano.

4.2.4. Resumo da arquitectura

Para uma melhor percepção do sistema e dos seus módulos constituintes, a Fig.23 mostra o diagrama de SBS (System Breakdown Structure).

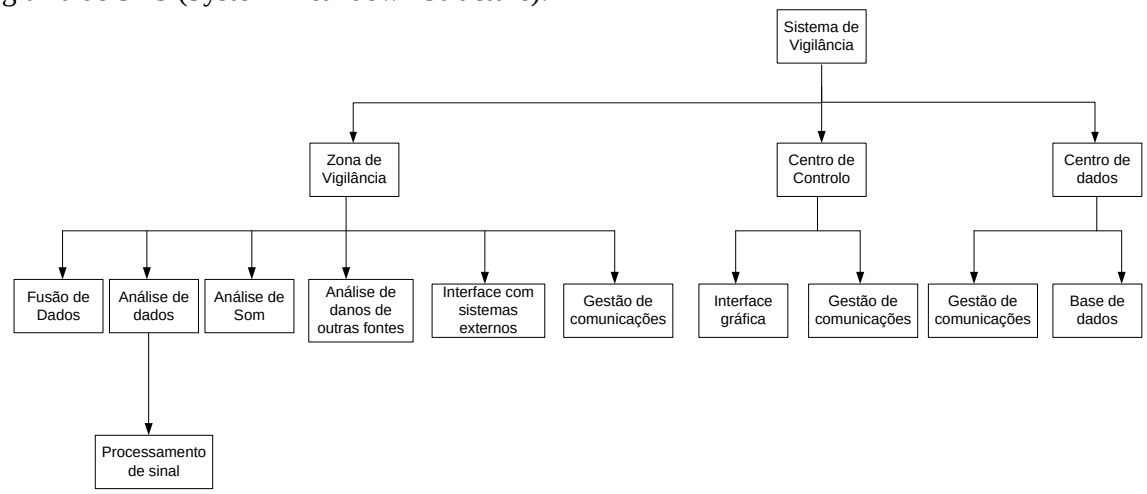


Fig. 23 - SBS do sistema.

O SBS mostra o sistema de vigilância dividido nas suas partes fundamentais. Através desta abordagem por blocos torna-se mais fácil perceber a organização do sistema bem como a interacção entre os diferentes blocos.

4.2.5. Interacção entre nós

Este sistema foi pensado de forma a adaptar-se a diferentes tipos de situações. Com o objectivo de tornar esta arquitectura robusta e flexível, dividiu-se o sistema em três partes como já foi especificado. A comunicação entre estas três partes pode ser feita através de diferentes alternativas.

4.2.5.1. Implementação através de uma LAN

Todo o sistema pode ser conectado através de uma LAN, tal como ilustra a Fig.24:

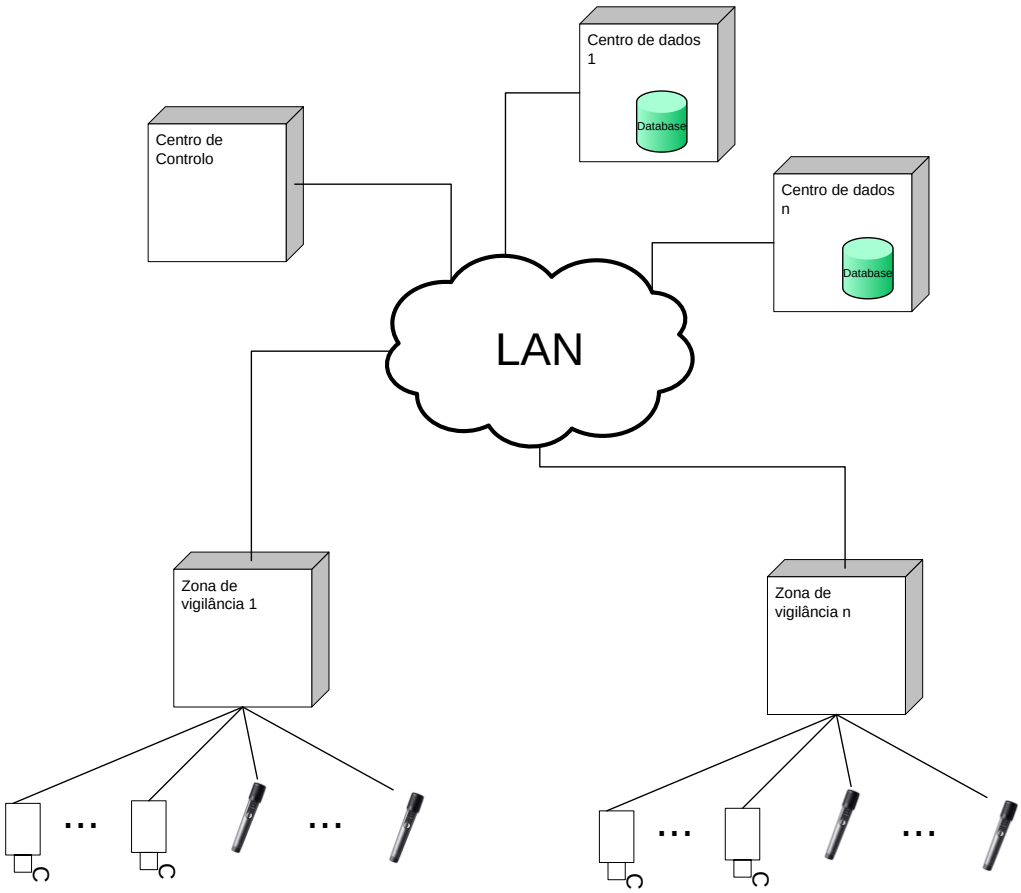


Fig. 24 - Implementação com LAN

Esta topologia encaixa em soluções cujas zonas a vigiar se encontram relativamente perto do centro de controlo e do centro de dados. Pode por exemplo ser aplicado num aeroporto ou num centro comercial.

4.2.5.2. Implementação através de internet

Outra alternativa reside em ligar as diferentes partes do sistema através da internet como é mostrado na Fig.25:

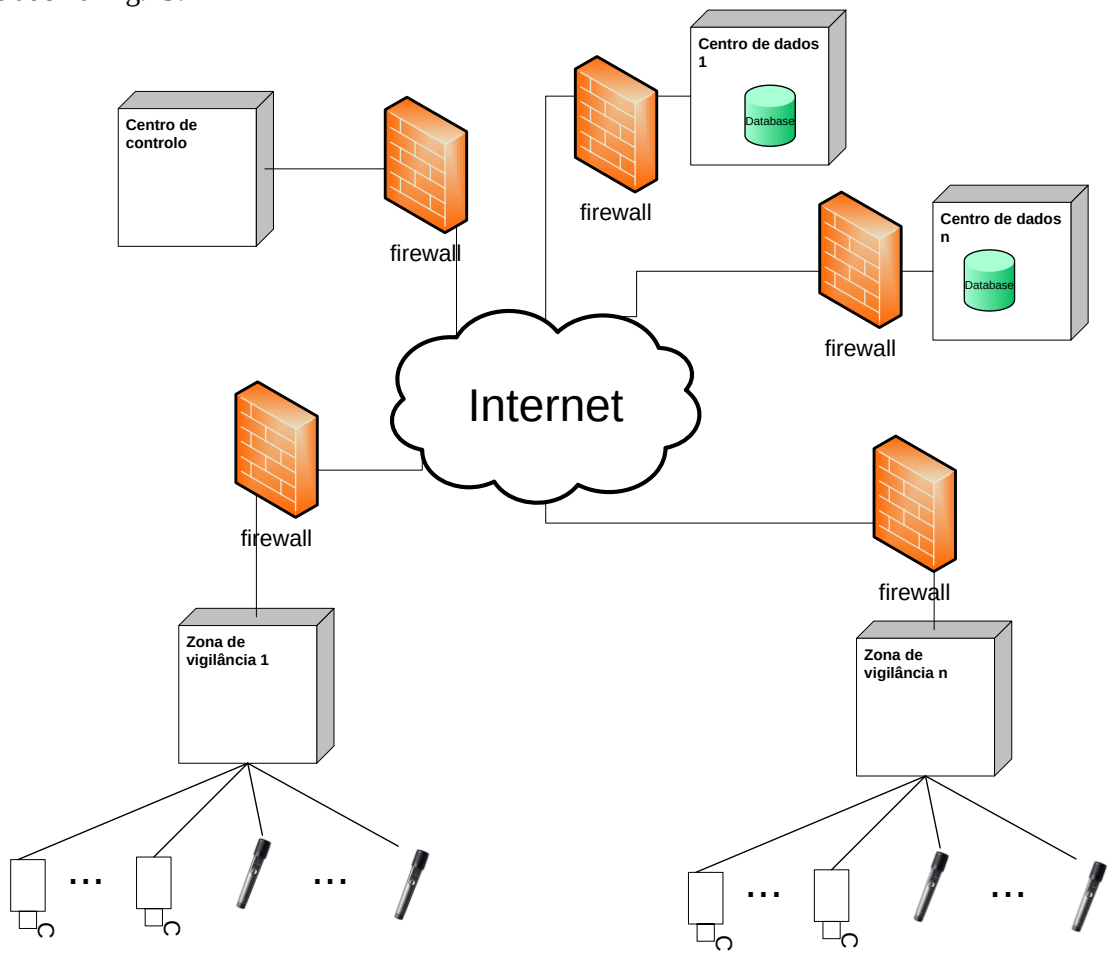


Fig. 25 -- Implementação com Internet.

Este cenário aplica-se nos casos em que ambos os nós estão separados fisicamente através de vários quilómetros de distância.

A *firewall* tem como objectivo proteger o sistema de vigilância e ataques externos, esta pode estar embebida num *router* ou até conjugada com outros sistemas de protecção para tornar a segurança do sistema eficaz.

4.2.5.3. Implementação através da internet e LAN

Apesar de ser possível o centro de dados estar separado fisicamente através da internet do centro de controlo, esta não é a solução mais eficaz, pois pode provocar atrasos na troca de informação entre o centro de controlo e o centro de dados. Assim, propõe-se implementação esquematizada na Fig.26:

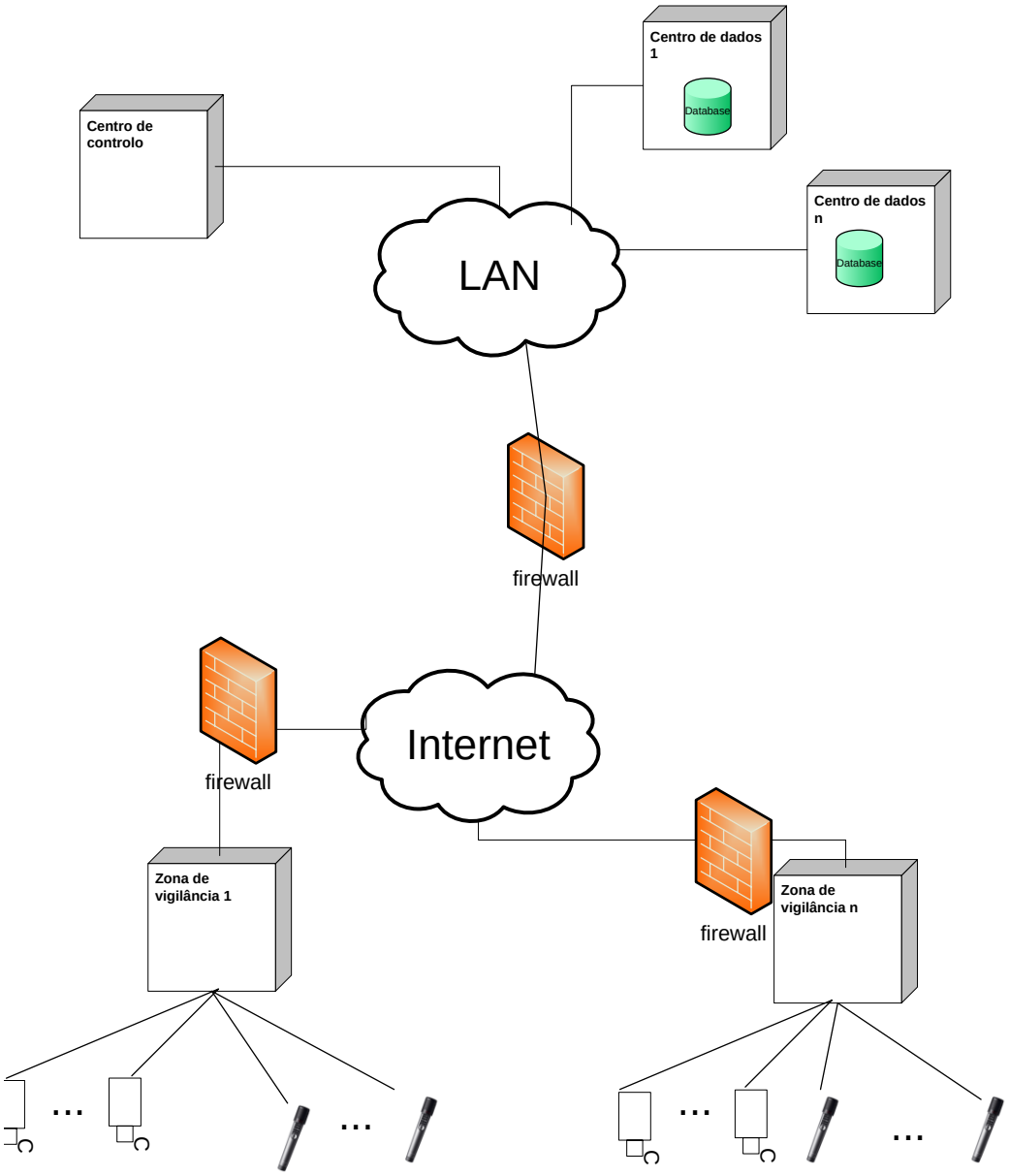


Fig. 26 - Implementação internet e LAN.

Como se pode observar, nesta topologia o centro de controlo e o centro de dados encontram-se na mesma LAN, enquanto que as zonas de vigilância se encontram fisicamente distantes, sendo apenas possível liga-las ao centro de controlo e ao centro de dados através da internet. Este tipo de configuração pode ser por exemplo aplicado em estações de metro, onde as várias estações a vigiar se encontram fisicamente distantes e existe um único centro de controlo.

Poderá também ser aplicada na vigilância de auto-estradas ou na vigilância de uma grande cidade.

Capítulo 5

5. Implementação

O desenvolvimento de um protótipo funcional foi um dos objectivos traçados para esta dissertação. O desenvolvimento deste protótipo foi baseado no *software* de vigilância livre ZoneMinder.Foi integrado neste *software* um algoritmo de *segmentação* e outro de *tracking*. Estes métodos são importantes na base dos sistemas inteligentes de vigilância.

Neste capítulo, primeiramente será detalhado o funcionamento do ZoneMinder, posteriormente serão abordados os algoritmos de *segmentação* e *tracking* e a forma como estes foram integrados no ZoneMinder.

5.1. ZoneMinder

O ZoneMinder é um *software* de vigilância livre que suporta diversas câmaras locais ou através da rede e possui um sofisticado sistema de detecção de movimento. É utilizado um algoritmo de detecção de movimento para gerar eventos automaticamente. Os eventos são guardados numa base de dados MySQL e podem ser visualizados imediatamente ou arquivados para visualização posterior. Possui uma interface web que permite controlar e gerir as funções remotamente. Também permite definir varias zonas para cada câmara, variando-se parâmetros como a sensibilidade e funcionalidade.

As funcionalidades apresentadas pelo ZoneMinder actualmente são um pouco limitadas, pois apenas se baseia na detecção de movimento. O objectivo no contexto desta dissertação é adicionar ao actual código de detecção de movimentos, um algoritmo de subtracção de fundo e outro de *segmentação*.

5.1.1. Requisitos

O ZoneMinder necessita de um conjunto de requisitos para funcionar. Assim, para os requisitos de *hardware* recomenda-se pelo menos 512MB de memória RAM e um disco externo com capacidade igual ou superior a 80GB pois a função DVR necessitará de espaço para alocar

as imagens gravadas e quanto maior for esse espaço, mais tempo de gravação poderá ser armazenado.

Uma vez que o ZoneMinder utiliza uma base de dados para armazenar a informação gerada quando detecta movimentos, é necessário instalar o MySQLl. Para a interface gráfica é usado o PHP, pois esta é baseada em Html e JavaScript. A infra-estrutura Web do sistema é suportada pelo Apache, MySQL e PHP.

Ao ZoneMinder é possível acoplar várias câmaras, para isso é necessário que estas sejam compatíveis, pode-se verificar a sua compatibilidade através de uma simples inspecção no Wiki (http://www.zoneminder.com/wiki/index.php/Supported_*hardware*)..

Para compilar e instalar o ZoneMinder, também é necessário instalar um conjunto de bibliotecas. Para desenvolver este trabalho, instalou-se no Ubuntu 7.10 a versão 1.23.3 do ZoneMinder que requereu os seguintes módulos:

- apache2
- php5
- php5-mysql
- libapache2-mod-auth-mysql
- mysql-server
- g++
- libssl-dev
- gnutls-dev
- libjpeg62-dev
- perl-byacc
- perl-debug
- perl-doc
- perl-ifeffit
- libclass-date-perl
- libdate-manip-perl
- PHP-Serialization-0.27

O ZoneMinder usa vários módulos de perl que para serem devidamente compilados e instalados necessitam de bibliotecas adicionais.

5.1.2. Arquitectura de *Software*

O ZoneMinder é constituído por numerosos componentes. Estes componentes são escritos em diferentes linguagens de programação consoante a sua função, nomeadamente C++/Perl e PHP. Os componentes em C++ são responsáveis pelo processamento das frames de vídeo recebidas; os módulos em perl permitem uma melhor eficiência de integração do ZoneMinder

com outros scripts e possibilitam a execução de tarefas externas à interface gráfica. Para a interface gráfica são usados módulos em PHP.

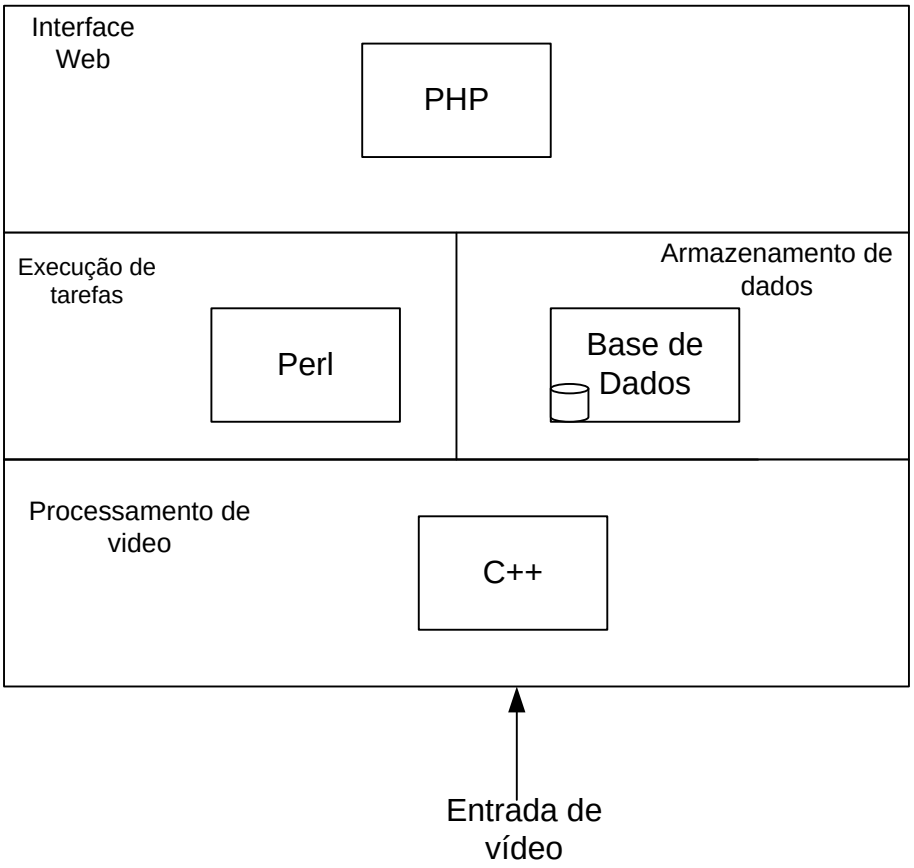


Fig. 27 - Integração do PHP, Perl e C++

Através da Fig.27 pode-se ter a ideia de como se relacionam os diferentes componentes construídos em diferentes linguagens. Assim, o processamento de sinal é feito através dos componentes construídos em C++ que recebem as frames de vídeo e guardam apenas aquelas em que existe movimento. A interface Web desenvolvida em PHP é o rosto do ZoneMinder. Quando um utilizador executa alguma acção através da interface Web como por exemplo iniciar ou parar o ZoneMinder, o que este faz é enviar para o script em Perl a acção a executar. A interface Web pode ainda ser usada para consultar directamente a base de dados e para exibir as imagens armazenadas.

5.1.2.1. Componentes

Para melhor perceber o conteúdo e o funcionamento do ZoneMinder, segue-se uma breve descrição dos principais componentes.

Módulos em C++:

- Zmc - ZoneMinder Capture deamon. Responsável por capturar as imagens de vídeo.
- Zma - ZoneMinder Analysis *daemon*. Este componente analisa as frames capturadas, verificando se existe movimento que possa gerar um evento ou alarme.

- Zms - ZoneMinder Streaming servidor. A interface Web conecta-se com este componente para obter imagens em tempo real ou consultar um histórico de imagens.

Módulos em Perl:

- Zmpkg.pl - ZoneMinder Package Control script. Controla a execução de todo o sistema.
- Zmcontrol.pl – É usado para controlar vários parâmetros das câmaras, como por exemplo o zoom ou o seu movimento. Este script converte as ordens e envia-as para a câmara segundo o protocolo de cada câmara. Caso a câmara não esteja na lista de câmaras compatíveis será necessário criar um novo script.

5.1.3. Arquitectura física

ZoneMinder apresenta uma arquitectura estruturada por um conjunto de componentes. O ZoneMinder para além das simples acções de videovigilância apresenta também um conjunto de opções variadas. Este permite que quando é gerado um alarme o utilizador seja notificado via SMS ou email. Outro dado relevante é a facilidade com que se pode configurar ZoneMinder para aceder via Web, ou seja, em qualquer lugar onde haja acesso à internet é possível conectar-se remotamente à interface do ZoneMinder para gerir ou consultar todo o sistema.

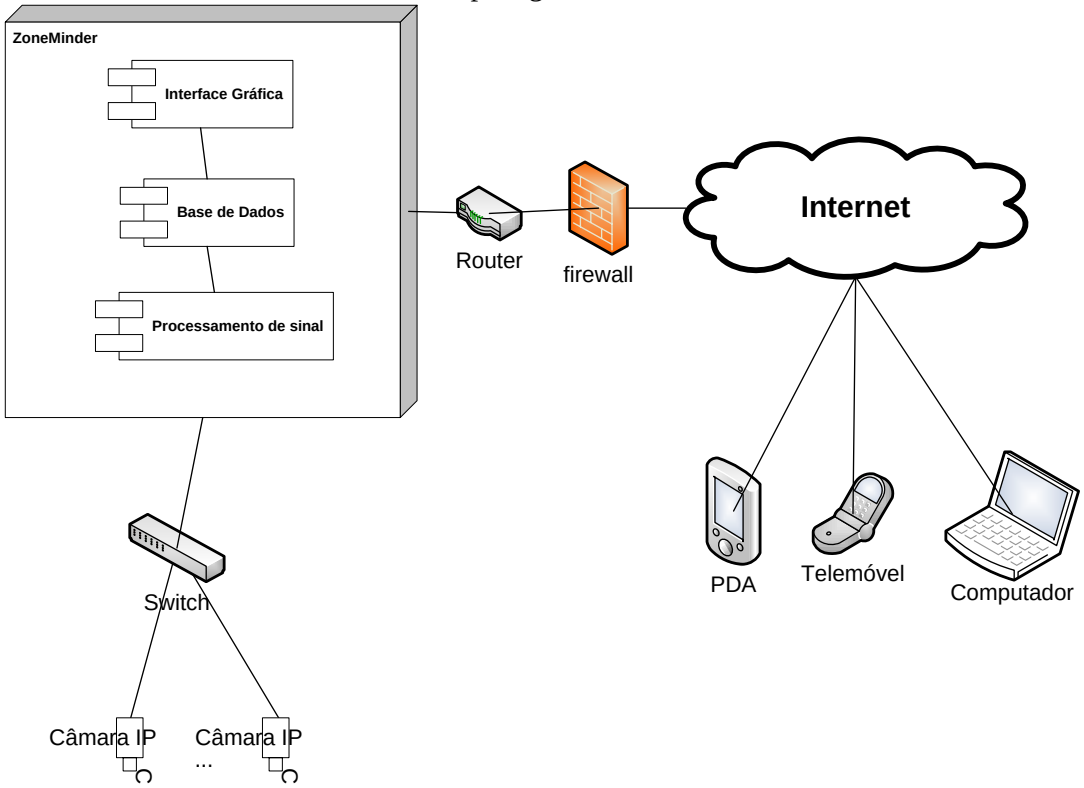


Fig. 28 - Arquitectura física

A Fig. 28 mostra uma possível arquitectura física que integra o ZoneMinder. Assim, a este podem estar conectadas *n* câmaras IP ou webcams. Depois, através da conexão à internet do ZoneMinder e posterior configuração é possível aceder ao sistema através de uma interface com

acesso à internet, como por exemplo um PC, telemóvel ou PDA. Para aumentar a segurança do sistema relativamente a possíveis ameaças provenientes da internet é acoplada uma *firewall*, que tradicionalmente já vem embebida com o *router*.

5.1.4. Interface gráfica

A interface com o utilizador é o rosto da aplicação. É a componente visual que permite ao utilizador introduzir e receber informação, visual ou textual.

O ZoneMinder apresenta uma interface gráfica simples, bem complementada com um conjunto de opções avançadas. Esta interface é acessível através da internet por um *browser*.

A Fig.29 mostra o diagrama de casos de utilização para melhor se perceber a interacção do utilizador com o ZoneMinder.

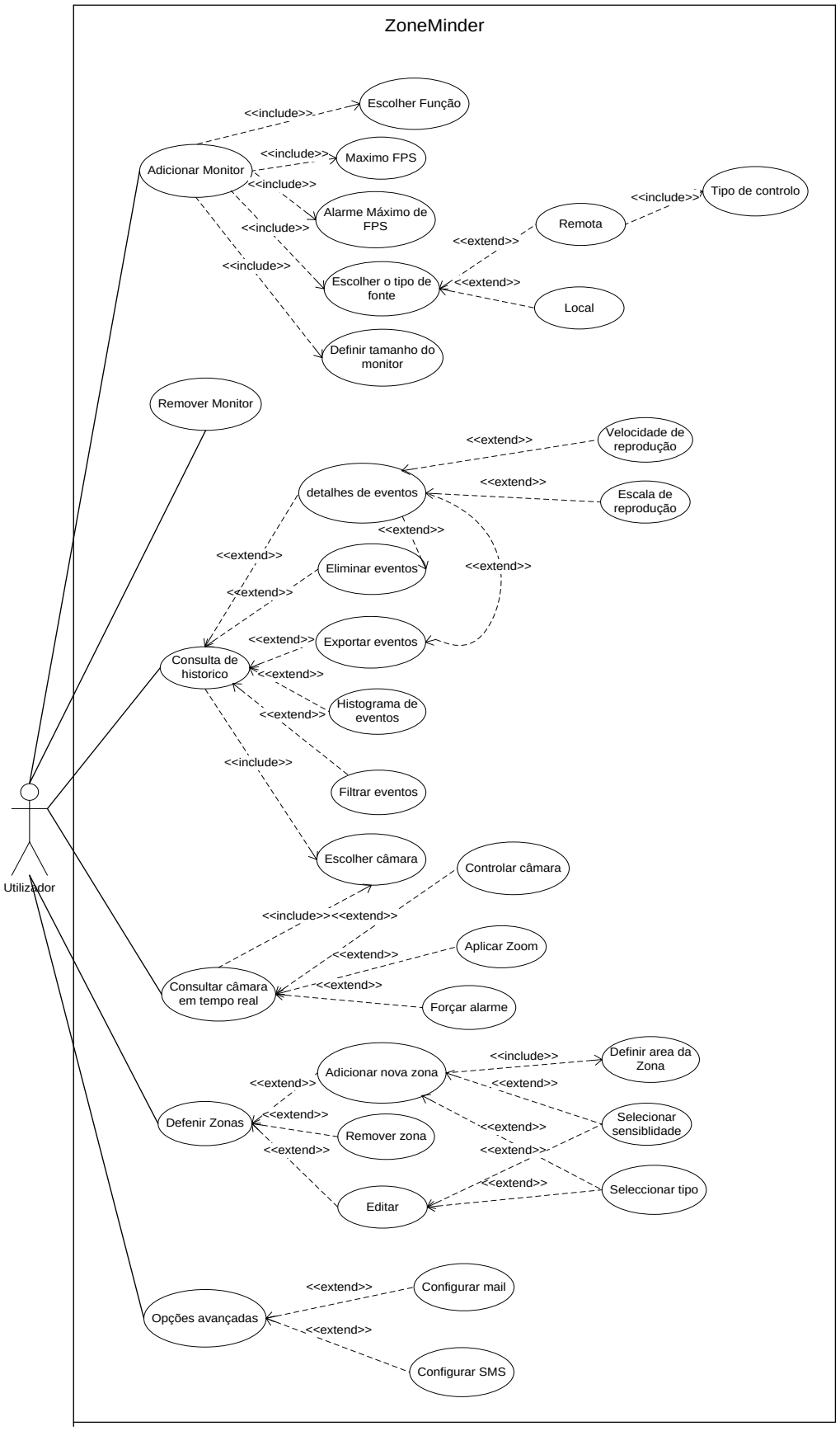


Fig. 29 - Diagrama de Casos de Utilização do ZoneMinder.

O diagrama de casos de utilização da Fig.29 mostra as principais opções que podem ser feitas através do utilizador. O utilizador tem à sua disposição a interface Web mostrada na Fig.30.

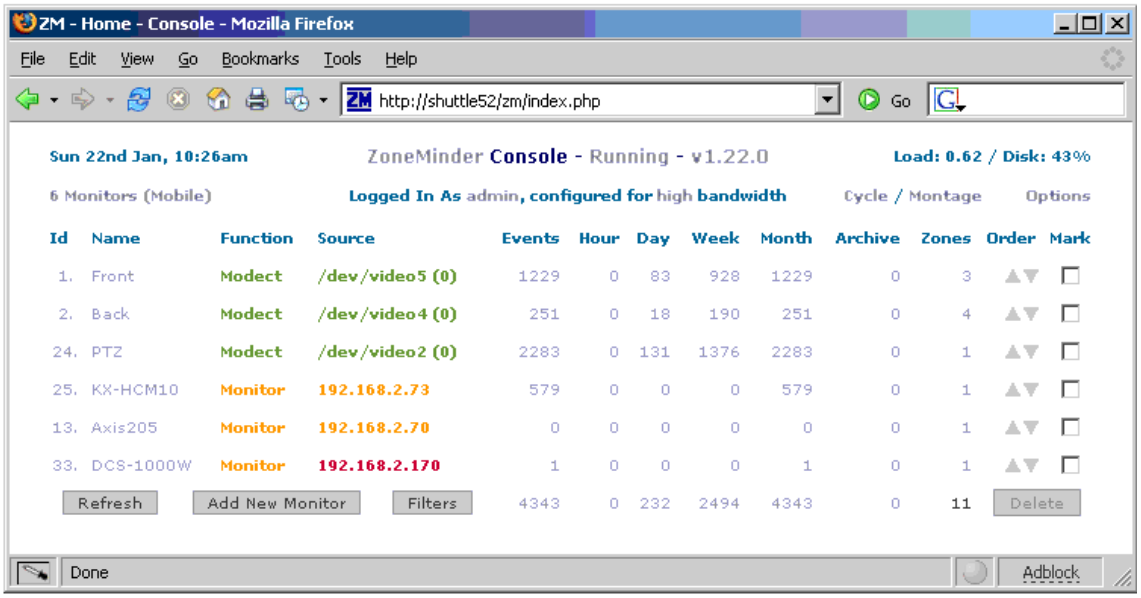


Fig. 30 - Consola do ZoneMinder

A interface Web é um conjunto de páginas Web dinâmicas, que permitem ao utilizador receber e enviar informação para todo o sistema. Assim, através desta consola que representa o rosto de ZoneMinder podem-se tomar várias opções, das quais se destacam as seguintes: i)adicionar monitor, ii)remover monitor, iii)consultar histórico, iv)consultar câmara em tempo real, v)definir zonas e vi)opções avançadas. Estas seis opções destacadas no diagrama da Fig.30 serão agora pormenorizadas.

5.1.4.1. Adicionar/Remover Monitor

Sendo o ZoneMinder uma aplicação que suporta diversas câmaras de vídeo, estas podem estar conectadas localmente ou remotamente. O ZoneMinder, através da sua interface gráfica, possibilita adicionar e remover monitores de forma bastante simples. Para adicionar um monitor ter-se-á necessariamente que especificar alguns parâmetros importantes. Ao clicar em "Add New Monitor" aparecerá a interface mostrada na Fig.31.

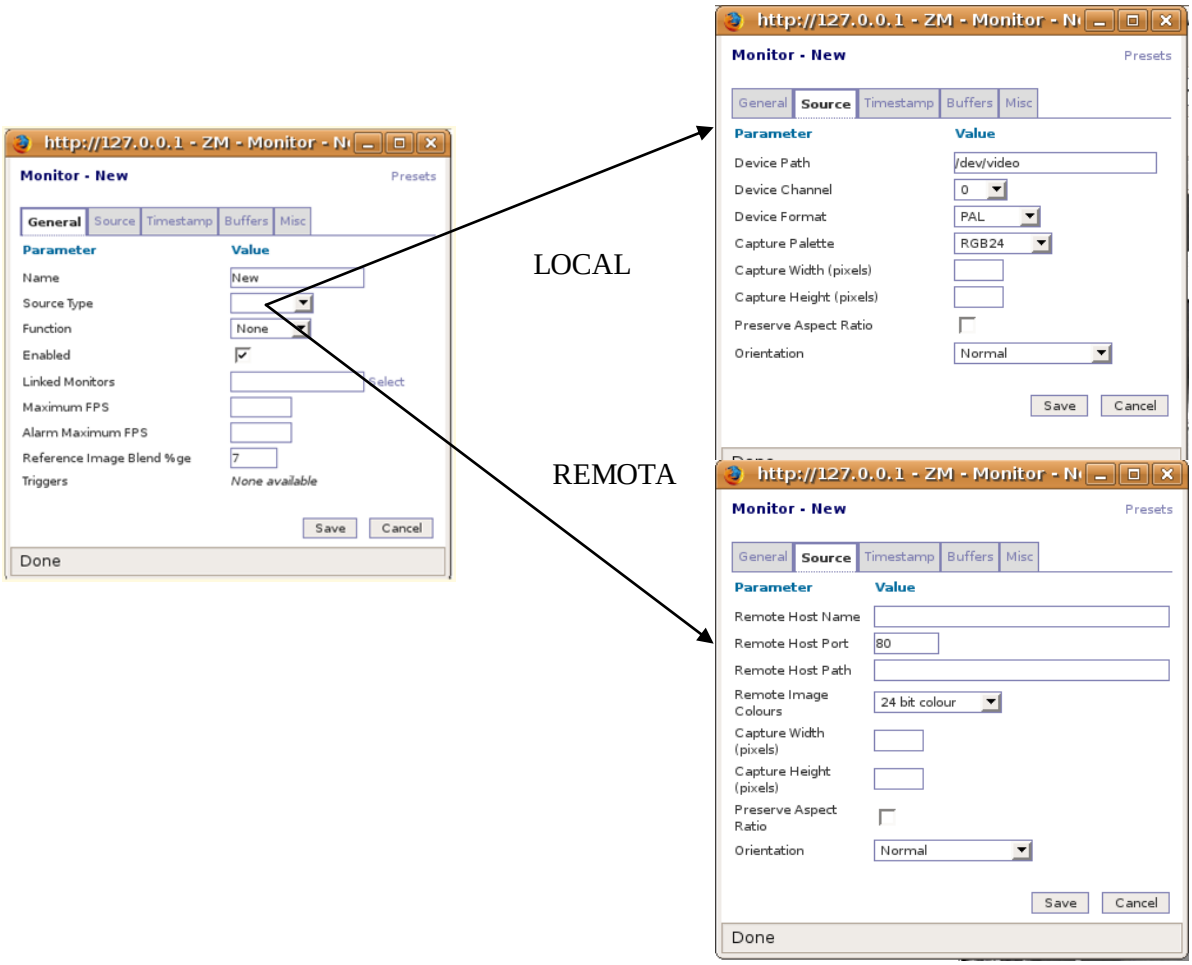


Fig. 31 - Adicionar Monitor.

Na interface gráfica da Fig. 31 é necessário especificar o nome do monitor, o *Source Type*, *Funtcion*, *Maximum FPS* e *Alarm Maximum FPS*.

O *Source Type* corresponde a especificar se o novo monitor a adicionar está conectado localmente ou remotamente. Consoante estas duas opções, o separador *Source* altera os seus campos, tal como é visível na Fig.31, para uma câmara local especifica-se o *Device Path* e para uma câmera IP coloca-se em *Remote Host Name* o endereço da câmara remota.

O campo *Function* define a função do monitor, que contém as seguintes opções:

- *None* - O monitor está desactivado e não gera eventos.
- *Monitor* – O monitor apenas transmitirá imagens para o utilizador monitorar, não fará análise dessas imagens nem gerará eventos ou alarmes.
- *Modect* – Corresponde ao modo "*Motion detection*", em que captura imagens, analisa-as e gera eventos caso seja detectado movimento.

- *Record* – Neste modo não processa a detecção de movimentos, mas são gravadas eventos contínuos de tamanho fixo independentemente do movimento registado na câmara.
- *Mocord* - Consiste numa mistura do modo *Modect* com *Record*, em que os resultados capturados no modo *Record* serão armazenados e estarão sujeitos à detecção de movimento podendo gerar eventos.
- *Nodect* - Significa "no *detection*". Modo especial para gravação de movimentos externos, isto é, só é gravado se houver movimento.

O campo "*Maximum FPS*(frames por segundo)" tem a função de limitar a taxa máxima de captura. Em algumas ocasiões podem existir uma ou mais câmaras capazes de capturar taxas elevadas, contudo nem sempre é necessário esse desempenho em todos os momentos é preferível aliviar a carga no servidor. Esta opção permite limitar a taxa máxima de captura para um determinado valor. Isso pode possibilitar ter mais câmaras suportadas pelo sistema através da redução de carga do CPU ou atribuir banda de vídeo desigual entre câmaras que partilham o mesmo dispositivo de vídeo. Esta opção controla o máximo de FPS, nas circunstâncias em que não é detectado alarme.

Outro campo que é obrigatório preencher é "*Alarm Maximum FPS*". Esta função configura o valor das frames por segundo que a câmara captura quando ocorre um evento.

Ainda no separador "*timestamp*" pode configurar-se em que posição da imagem aparece o tempo.

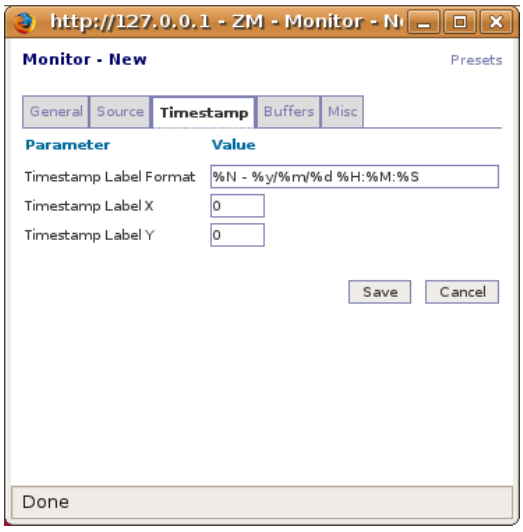


Fig. 32 - Separador *timestamp*.

- *Timestamp Label Format* – Corresponde ao formato do tempo. (*%N* = nome do monitor; *%y* = ano; *%m* = mês; *%d* = dia; *%H* = hora; *%M* = minuto; *%S* = segundos).
- *Timestamp Label X* - posição da etiqueta da esquerda para direita em pixéis.
- *Timestamp Label Y* - posição da etiqueta do topo para baixo em pixéis.

Terminadas todas as configurações e após clicar em *save*, estará adicionado o novo monitor, que funcionará conforme o especificado.

Na consola (Fig.30) pode-se ver um conjunto de informação útil a cada monitor, ver se este está activo, consultar quantos eventos detectou até ao momento entre outros. Para remover um ou mais monitores basta selecciona-los na consola e clicar em *Delete*.

Caso a câmara a utilizar suporte controlo PTZ (Pan ,Tilt e Zoom), será necessário escolher o tipo de controlo usado pela câmara. Para isso é indispensável a compatibilidade da câmara com o ZoneMinder. Caso esta não seja compatível, será necessário criar um novo script para que as funções de controlo funcionem.

5.1.4.2. Consultar histórico

O histórico de eventos detectados corresponde a uma lista detalhada dos eventos detectados em cada câmara. Para termos acesso a esse histórico deve-se escolher a câmara na qual se pretende consultar o histórico.

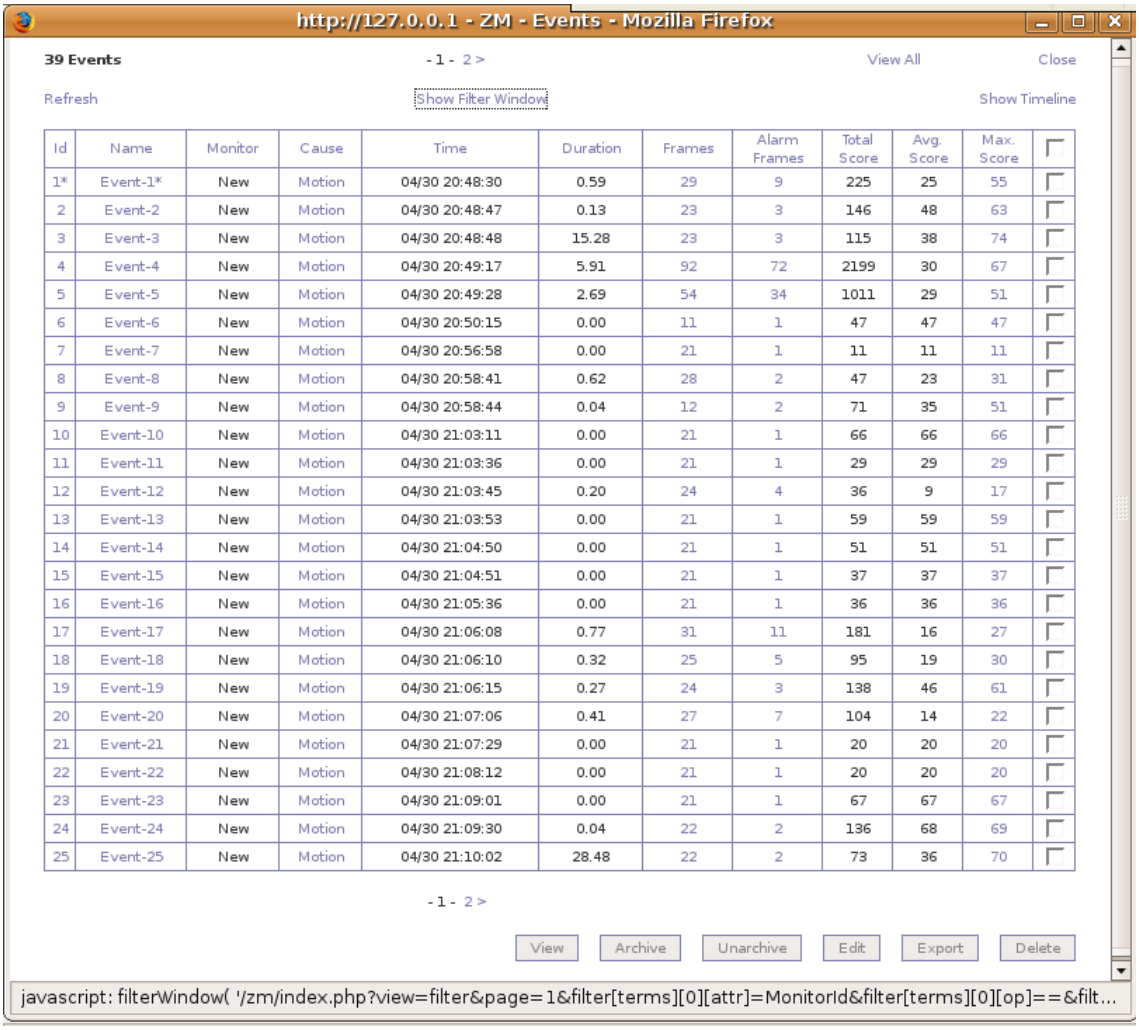


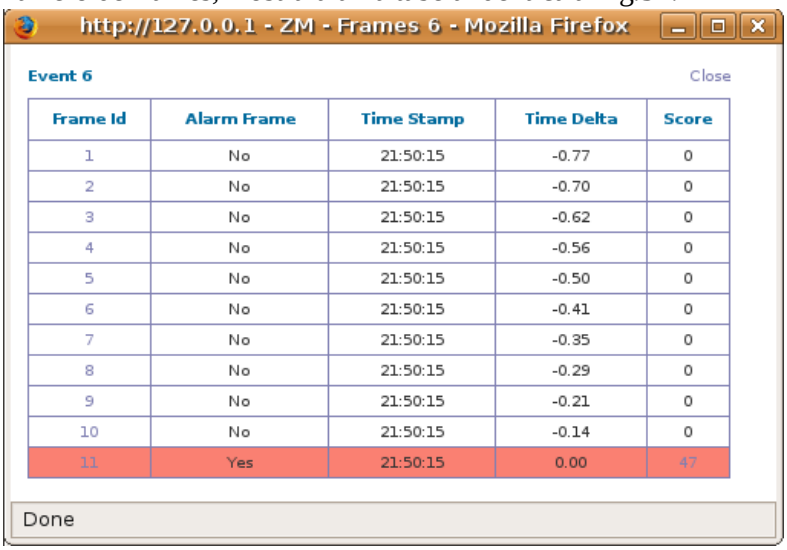
Fig. 33 - Histórico de eventos

A Fig.33 mostra a interface gráfica que lista um conjunto de eventos detectados por uma câmara. Nesta interface estão disponíveis várias opções. Pode-se filtrar os eventos para navegar

facilmente na lista de eventos que contém um breve resumo de cada evento detectado, assim como a sua duração, o número de frames, as frames em que foi detectado alarme e o *score*.

O *score* é um valor arbitrário que representa essencialmente a percentagem de pixéis de alarme numa determinada zona. Este *score* permite dar uma simples indicação de como o evento detectado foi importante.

Ao clicar no número de frames, mostrará uma tabela idêntica à Fig.34.



Frame Id	Alarm Frame	Time Stamp	Time Delta	Score
1	No	21:50:15	-0.77	0
2	No	21:50:15	-0.70	0
3	No	21:50:15	-0.62	0
4	No	21:50:15	-0.56	0
5	No	21:50:15	-0.50	0
6	No	21:50:15	-0.41	0
7	No	21:50:15	-0.35	0
8	No	21:50:15	-0.29	0
9	No	21:50:15	-0.21	0
10	No	21:50:15	-0.14	0
11	Yes	21:50:15	0.00	47

Fig. 34 - Lista de Frames detectados

A tabela da Fig.34 mostra em detalhe todos as frames capturados pela câmara. As frames em que sejam detectados alarmes apresentam uma cor vermelha explicitando o respectivo *score*.

A interface de eventos apresentada na Fig.33 também possibilita eliminar os eventos que não interessam. Para isso basta seleccioná-los e remove-los. Da mesma maneira é possível exportar os eventos, caso seja necessário dar-lhes outro uso. Quando clicado sobre um determinado evento apresentar-se-á a interface da Fig.35. Através desta interface é possível ver e rever o evento, podendo reproduzir-se o vídeo com maior ou menor velocidade e aumentar a imagem para uma análise mais detalhada.

Fig. 35 - Interface gráfica para consulta de um evento

Na lista de eventos (Fig.33) existe ainda possibilidade de se ver um gráfico dos eventos detectados ao longo do tempo, como mostra a Fig.36.

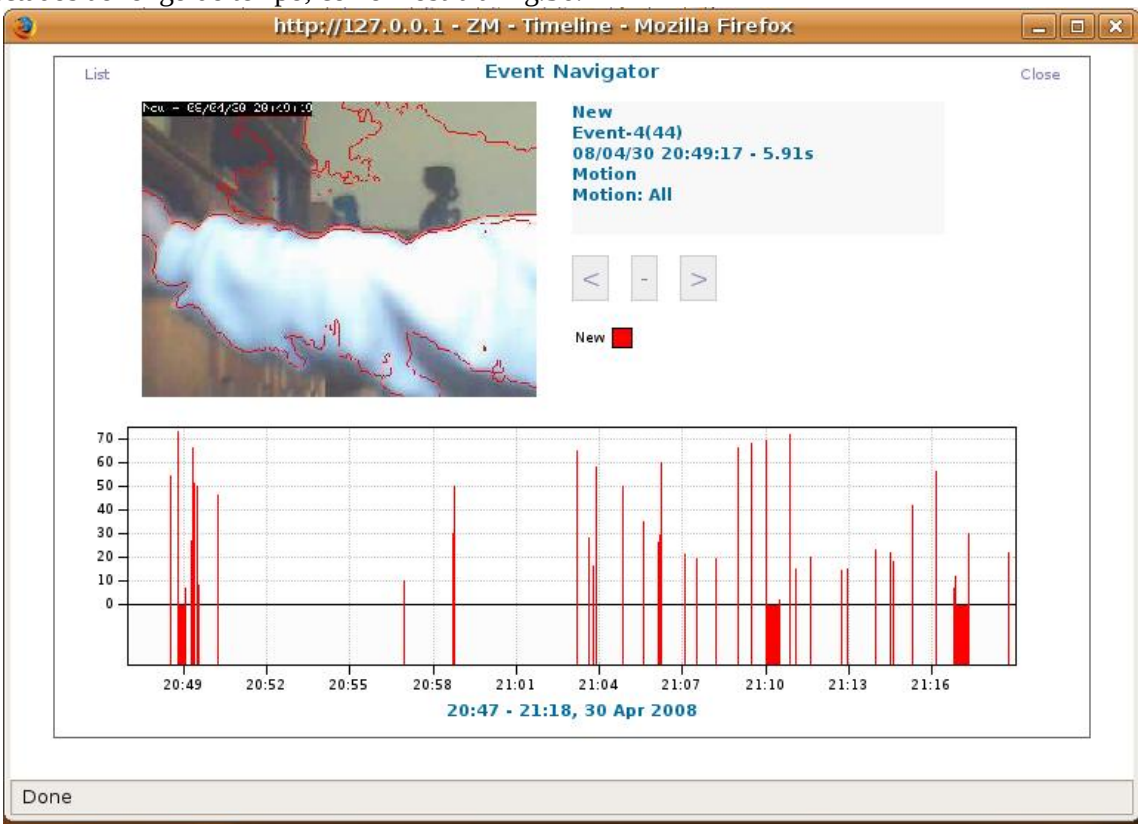


Fig. 36 - Gráfico da ocorrência de eventos ao longo do tempo

Este gráfico apresenta o volume da ocorrência de eventos ao longo do tempo. É possível clicar sobre o gráfico e aceder ao detalhe dos eventos.

5.1.4.3. Consultar câmara em tempo real

O ZoneMinder tem a capacidade de gerar eventos e processar o sinal de entrada e ao mesmo tempo transmitir a imagem de uma ou mais câmaras em tempo real para o operador humano consultar.

Assim, quando na consola (Fig.30) se clica no monitor a visualizar será apresentado a interface da Fig.37.

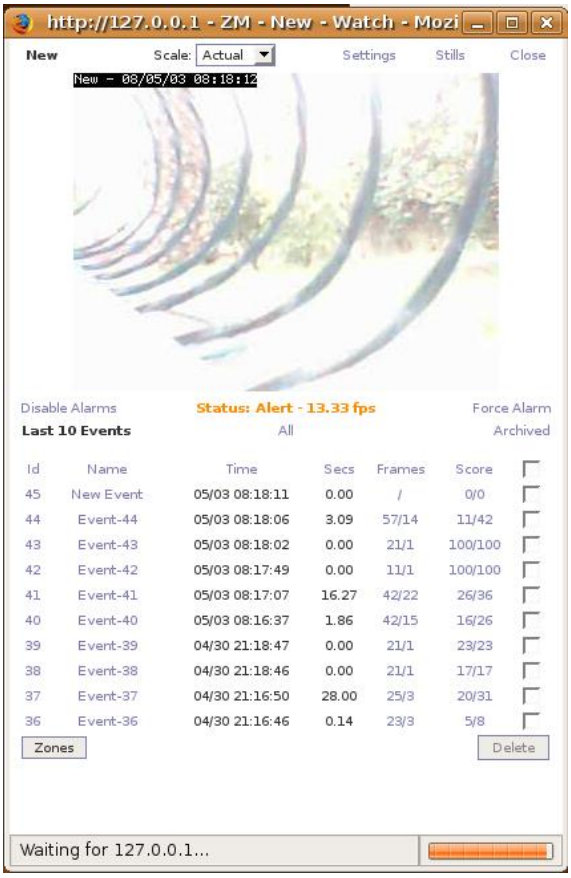


Fig. 37 - Interface visual com câmara.

Na interface da câmara local, é mostranda a imagem em tempo real, bem como os eventos detectados por essa câmara até ao momento. Uma importante tarefa que é possível fazer nesta interface é forçar um alarme, pois se num determinado momento o operador deseja registar um dado acontecimento, este pode forçar um alarme para registar esse acontecimento.

5.1.4.4. Definir Zonas

Esta é uma das tarefas mais importantes e úteis do ZoneMinder, pois possibilita que sejam gerados eventos ou alarmes numa zona específica da imagem seleccionada pelo utilizador. A cada câmara pode se associada uma ou mais zonas, com diferentes sensibilidades.

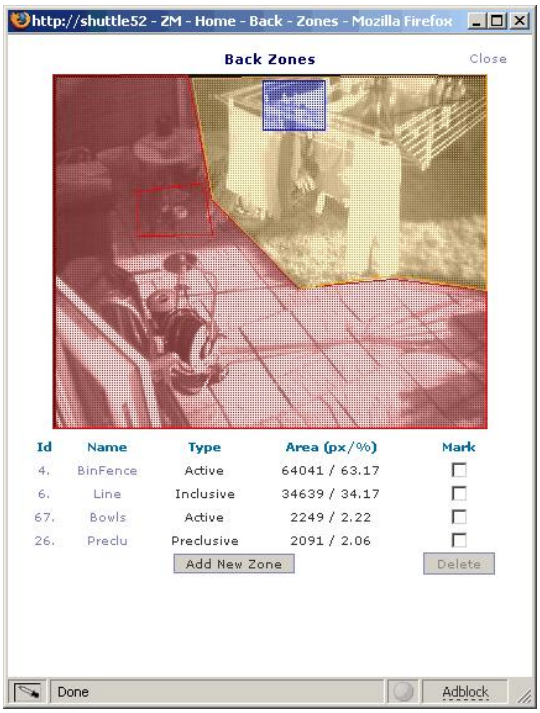


Fig. 38 - Interface para adicionar uma nova zona

A interface da Fig.39 mostra a lista das zonas existentes. Através desta interface é possível eliminar e editar uma dada zona ou adicionar uma nova zona clicando em "*Add New Zone* ". Onde constam as opções da Fig.40. Através desta interface pode-se seleccionar a área da zona em que se deseja registar os eventos. Depois é necessário configurar alguns parâmetros.

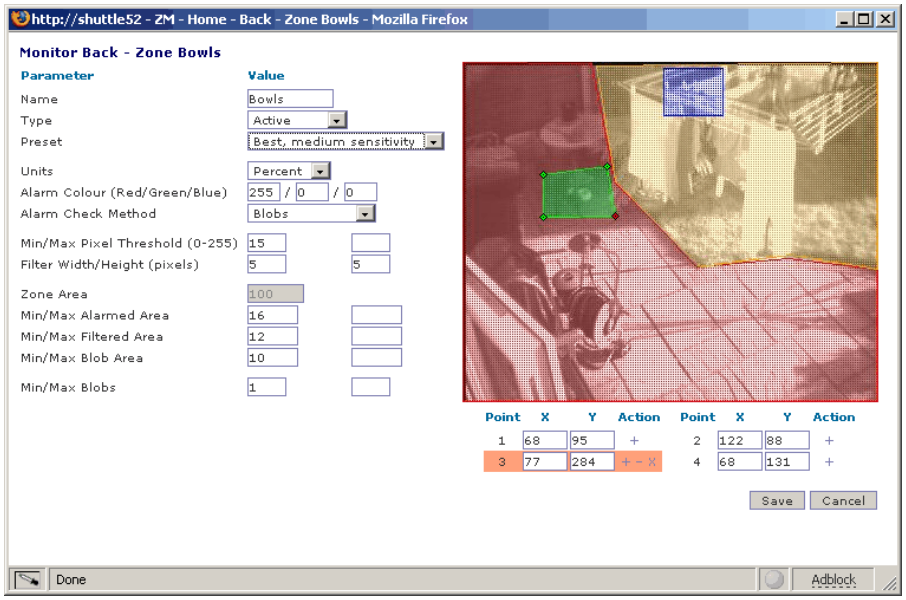


Fig. 39 - Adicionar ou editar Zonas.

O tipo de zona é um dos parâmetros mais importantes, para o qual se tem as seguintes cinco opções:

- **Active** – Corresponde a um dos tipos de zonas usados mais frequentemente. Este acciona um alarme sobre quaisquer acontecimentos que ocorrem no interior da zona definida.
- **Inclusive** – Este tipo de zona acciona um alarme se pelo menos um outro alarme já foi accionado noutra zona. Este tipo de zona é útil para locais com, por exemplo, plantas, que se movem muito e que iriam accionar muitos alarmes. Assim, se for necessário cobrir algo que está por detrás de uma planta, podem associar-se uma zona inclusive na área da planta e outra zona activa no local onde se deseja vigiar. Apenas ocorrerá alarme quando for detectado movimento nas duas zonas.
- **Exclusive** – Os alarmes destas zonas apenas serão accionados se não houver alarmes detectados nas zonas activas.
- **Preclusive** – Os movimentos ou outras mudanças que ocorrem nas zonas Preclusive terão o efeito de assegurar que não ocorrerá novamente em todas as outras zonas. Este tipo de zona é usada para combater os alarmes formados pela mudança de iluminação e outros ajustes de imagem. Geralmente isto pode ser conseguido através da limitação do número máximo de pixéis de alarme ou outra medida de uma zona activa. No entanto, em alguns casos, essa zona pode cobrir uma área onde a área de iluminação variável ocorre em diversos locais como o sol ou movimentos das sombras e, assim, pode ser difícil chegar a valores gerais. Estas zonas devem estar situadas em áreas da imagem que têm menos probabilidade de haver movimento. Deve ocorrer uma mudança geral de iluminação que seria desencadeada pelo menos tão cedo quanto qualquer uma zona activa e impedir quaisquer outras zonas de gerar um alarme.
- **Inactive** – Nesta zona acontece exactamente o contrário da zona *active*. Neste tipo de zonas a ocorrência de alarmes não serão notificadas. Serve para cobrir zonas em que não ocorrerão eventos relevantes, prevenindo assim falsos alarmes.

5.2. Algoritmo de *Segmentação*

A *segmentação* trata-se do primeiro passo na análise de imagens, consiste em usar um algoritmo para definir na imagem, contornos em redor de objectos de interesse. Os algoritmos de *segmentação* permitem encontrar diferenças entre dois ou mais objectos e distinguir os pixéis uns dos outros e do fundo. Esta distinção permitirá interpretar pixéis contíguos e agrupá-los em regiões.

O algoritmo utilizado para a introdução de uma tarefa de *segmentação* no ZoneMinder é explicitado no documento Teixeira et al.(2007).

5.2.1. Object Segmentation Using Background Modelling and Cascaded Change Detection

O objectivo deste algoritmo, visa separar os objectos contidos no cenário da forma mais eficaz e robusta. Para além das aplicações típicas da *segmentação*, tais como videovigilância, também são introduzidos mais dois requisitos adicionais: (1) deverá representar apenas uma pequena fracção do total do tempo de processamento e (2) processamento global em tempo real.

5.2.1.1. Algoritmo

O algoritmo a ser implementado é construído de forma a obter melhores resultados quando houver mudanças de iluminação ou ruído, adaptando-se automaticamente a essas mudanças. Também suporta identificação do comportamento dinâmico do fundo que se revela importante quando os objectos em movimento que ocupam uma pequena fracção de todo o campo de visão.

A Fig.40 mostra o diagrama de blocos do CMoG, onde mostra as etapas que a frame currente (F_c) passa até se obter o resultado final (S_c).

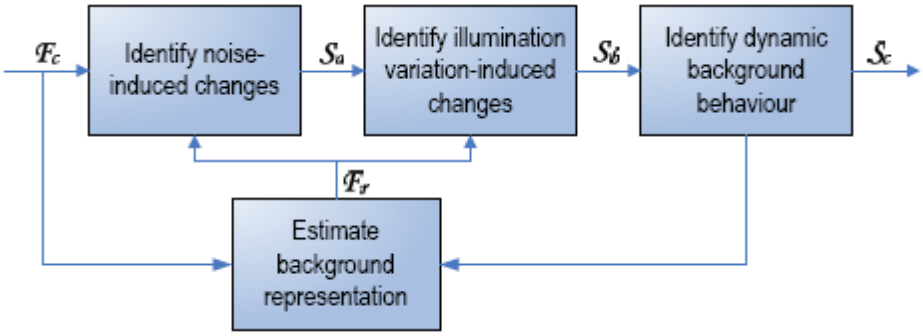


Fig. 40 - Diagrama de blocos do CMoG

O primeiro teste identifica o conjunto de pixéis S_a que não foram alterados devido a algum tipo de ruído. O bloco seguinte faz o mesmo processo para variâncias de iluminação (S_b). Por último eliminam-se mudanças estruturais que resultam do comportamento dinâmico do fundo, obtendo-se assim S_c. No final é actualizada a F_r (reference frame), importante para os primeiros dois processos.

$$P\left(\Delta^2 > \Delta^2_{(i,j)} \middle| H_0\right) = \frac{\prod\left(\frac{n}{2}, \frac{\Delta^2_{(i,j)}}{2\sigma_c^2}\right)}{\prod_2^n} \quad (6)$$

onde Δ^2, corresponde a uma variável aleatória.

A equação (6) reapresenta a possibilidade de um pixel sujeito a ruído pertencer ou não ao fundo, esta equação define o primeiro bloco do diagrama.

O segundo bloco é definido pela equação (7) que corresponde à possibilidade de um pixel sujeito a mudanças de iluminação pertencer ou não ao fundo.

$$cos\theta = \frac{v^c . v^r}{||v^c|| . ||v^r||} \quad (7)$$

onde v^ccorresponde ao valor do vector de cores pixel corrente e v^r é o vector de cores do pixel de referência.

$$P(B|v_t) = \frac{\sum_{k=1}^{K} P(v_t|G_k).P(G_k).P(B|Gk)}{\sum_{k=1}^{K} P(v_t|G_k).P(G_k)} \quad (8)$$

A equação (8) representa a probabilidade de um pixel pertencer ao fundo dinâmico, onde v_t é o vector de cores de cada pixel e G_k é o *k* termo da *K* distribuição Gaussiana.

5.3. Algoritmo de *tracking*

Tal como já foi referido no capítulo 2, o *tracking* consiste em seguir um ou mais objectos através de frames sucessivas para determinar como estes se movem na cena e relativamente a outros objectos.

Neste capítulo, será efectuado uma breve descrição acerca do algoritmo de *tracking* a integrar no ZoneMinder em conjunto com o algoritmo de *segmentação* já descrito. Este algoritmo é detalhado em Teixeira et al. (2008)

A constituição deste pode dividido em duas fases: detecção, onde a forma e os modelos da câmara são usados para a "análise de fronteira dos objectos *foreground"* para estimar hipóteses humanas; *tracking*, onde cada hipótese humana é monitorada nos frames consequentes usando um filtro de Kalman e as características do objecto. O algoritmo de *tracking* foi estruturado para que este seja computacionalmente mais eficiente

A câmara deve estar situada alguns metros acima do solo, o que permite fixar as dimensões médias de um humano e evita alguns problemas tais como: a distância do objecto em direcção à câmara; ocultação temporária ou parcial.

5.4. Implementação do protótipo

Após o levantamento dos requisitos e a definição da arquitectura, efectuados no capítulo4, foi iniciada a implementação de um protótipo, que tem como objectivo implementar parte da arquitectura concebida. Pretende-se que este seja uma base de desenvolvimento e teste de novos algoritmos e técnicas inteligentes para a vigilância.

Como já foi descrito, esta implementação consiste num upgrade ao *software* ZoneMinder. O *software* ZoneMinder é uma aplicação livre de fácil configuração e apresenta muitas. Contudo este ainda se encontra pouco desenvolvido no que diz respeito às técnicas de vigilância de terceira geração. O ZoneMinder apenas tem um bloco de detecção de movimento, e será adiccionado mais um bloco para a *segmentação* e o *tracking* de pessoas, devolvendo e armazenando no histórico o número de pessoas detectadas numa câmara.

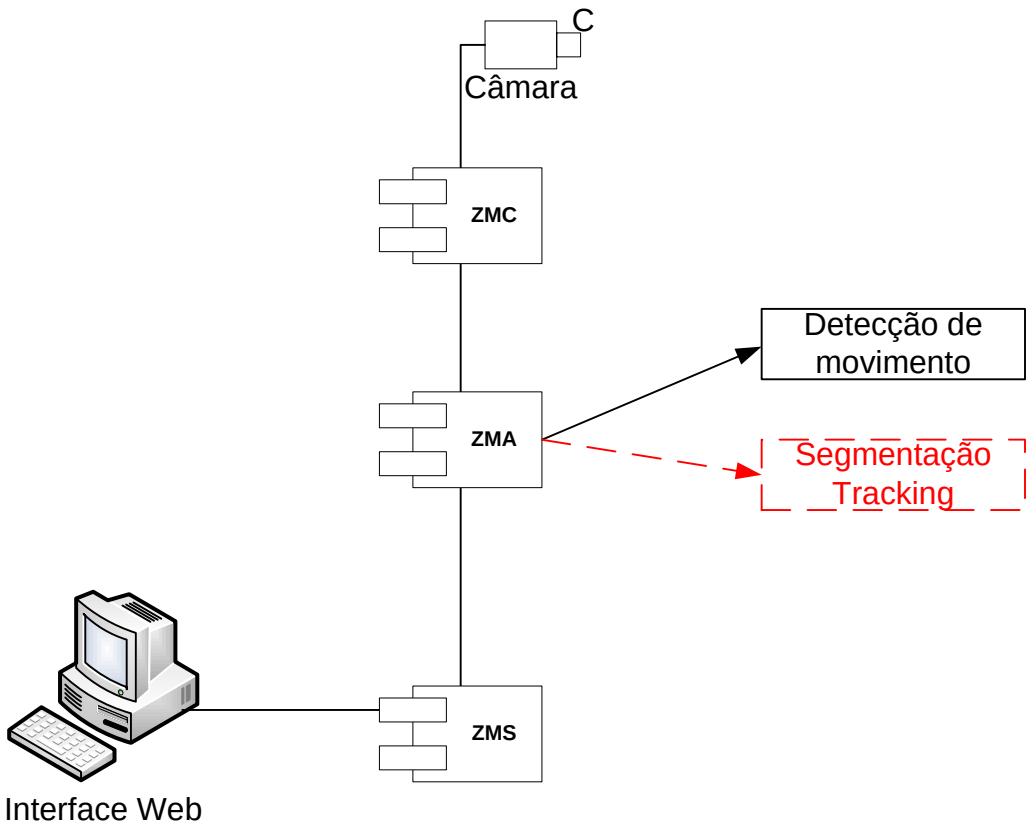


Fig. 41- Diagrama de componentes da implementação.

Na Fig.41 é mostrado o diagrama de componentes do ZoneMinder com o novo bloco adicionado. As imagens são capturadas pelo *daemon* ZMC (ZoneMinder Capture), de seguida são analisadas pelo *daemon* ZMA (ZoneMinder Analysis). O ZMS (ZoneMInder Streaming) tem como função enviar para a interface as imagens em tempo-real ou armazenadas no histórico.

Como podemos observar através do diagrama, o ZoneMinder apenas possui no interior do ZMA um bloco de *detecção de movimento*, com o desenvolvimento desta dissertação, este possui agora o bloco de *segmentação* e *tracking* referenciado a vermelho no diagrama.

Como perspectiva de implementação futura, com este trabalho abriram-se portas para a integração de mais módulos, como por exemplo *detecção de faces*, *análise comportamental* ou outro *tracking multi-câmara*.

Ainda de forma a fazer convergir este protótipo para arquitectura apresentada no capítulo 4, também se poderia criar um componente responsável pela *análise de áudio*, componente este que receberia, analisaria e armazenaria na base de dados. (Fig.42)

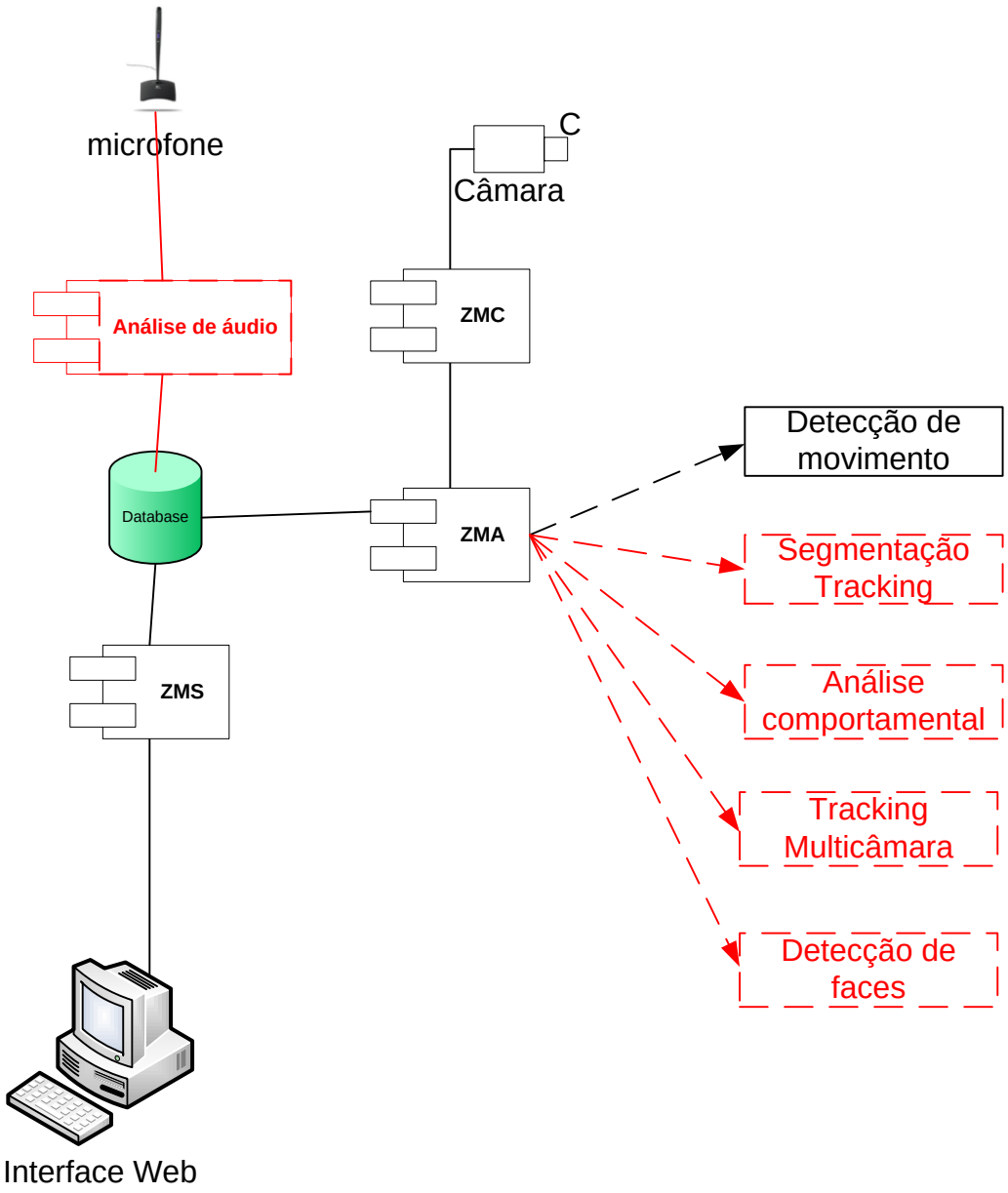


Fig. 42- Diagrama de componentes do trabalho a desenvolver no futuro.

5.5. Resultados

Após o levantamento dos requisitos e a definição da arquitectura, efectuados no capítulo anterior, foi feito o desenvolvimento da aplicação. Para tal, primeiramente começou-se por estudar o funcionamento e organização do ZoneMinder.

Depois de se integrar o algoritmo de *tracking* e de *segmentação* tal como demonstrado na Fig.41, procedeu-se à análise dos resultados.

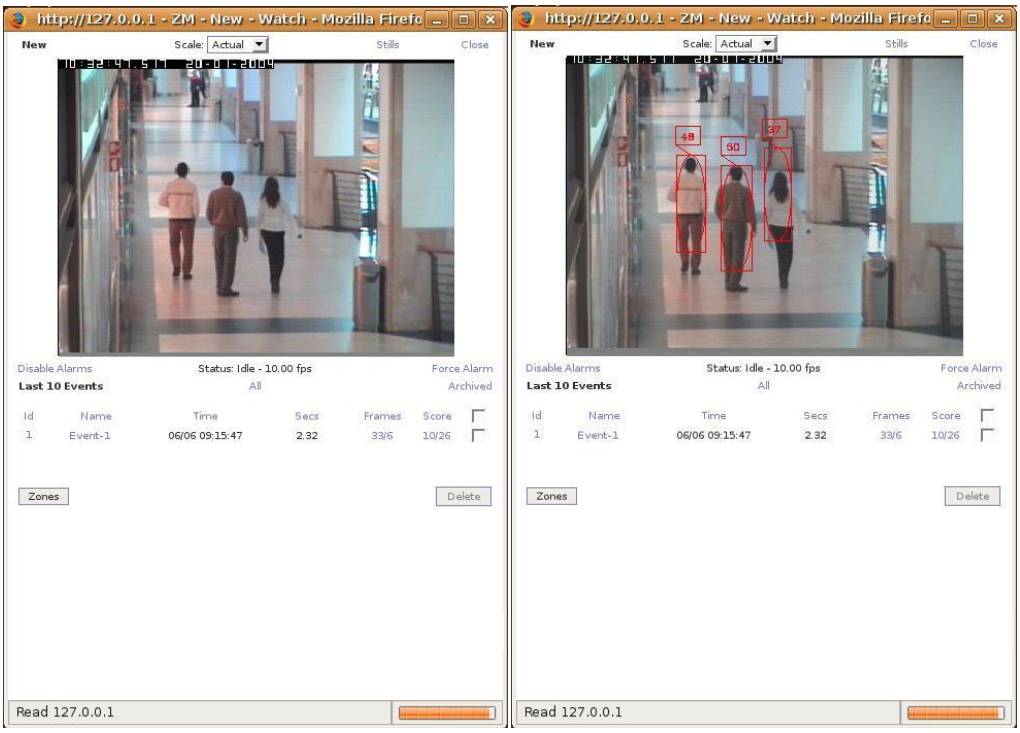


Fig. 43- À esquerda, são mostrados os eventos gerados apenas pelo ZoneMinder; à direita mostram-se os eventos gerados depois da integração dos algoritmos de *segmentação* e *tracking*.

Para testar o ZoneMinder, utilizou-se uma sequência pré-gravada. A Fig.43 mostra a diferença antes e depois da integração. Depois da implementação, os eventos gerados armazenam na base de dados o número de pessoas detectadas, bem como as frames resultantes do processo de *segmentação* e *tracking*, como demonstra a Fig.43 da direita. Nessas frames é possível verificar a identificação atribuída a cada pessoa detectada, através de um rectângulo envolvente.

Com esta implementação é possível saber o número de pessoas que é detectado em cada câmara. Assim, o aspecto da tabela de eventos também se alterou, como se pode constatar na Fig.44.

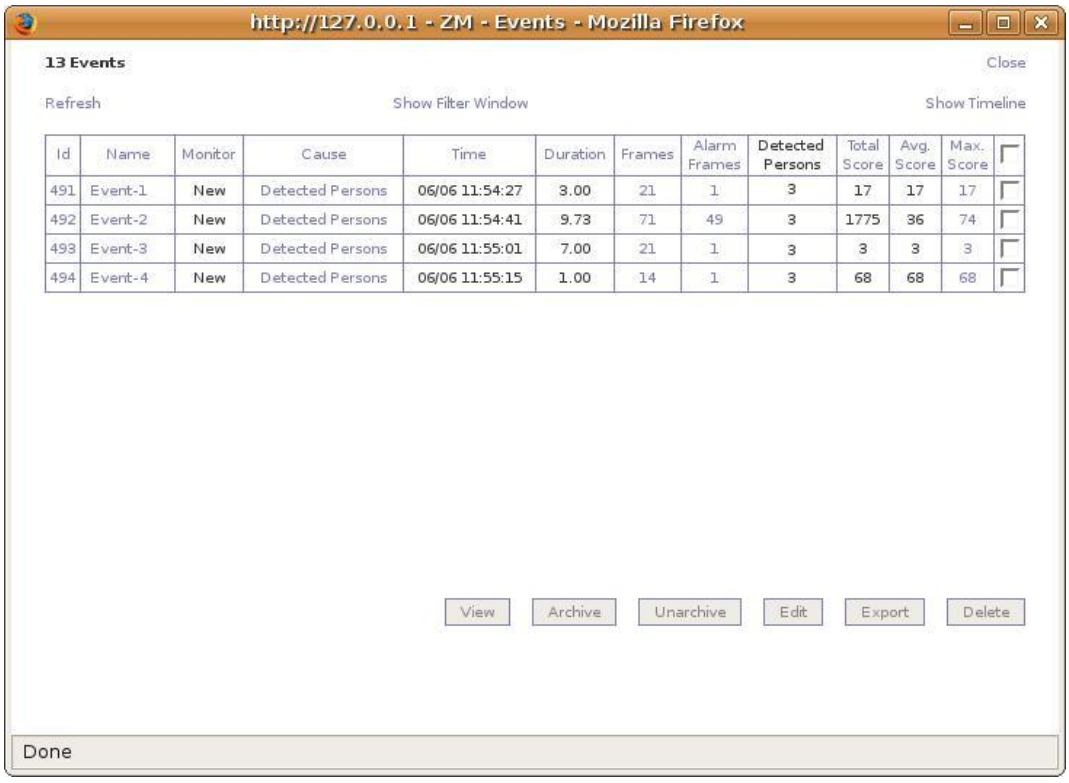


Fig. 44 - Tabela de eventos com o informação relativa ao número de pessoas detectadas..

Quando um utilizador quiser consultar a tabela de eventos de uma câmara, esta agora conterá uma nova coluna, com o nome de "Detected Persons". Nesta coluna é mostrado o número de pessoas detectadas em cada evento. Outra diferença é a descrição da causa do evento que agora também se designa "Detected Persons", que permite distinguir o modo de detecção que se está a usar.

5.5.1. Análise de desempenho

A implementação dos algoritmos de *segmentação* e *tracking* no ZoneMinder, tiverem um efeito um pouco negativo no que diz respeito ao desempenho do sistema. Pois, no modo *Detected Persons*, o ZoneMinder apenas consegue analisar cerca de 5 frames por segundo. Este comportamento deve-se aos algoritmos de segmentação e *tracking* cuja implementação não-optimizada ainda não suporta *frame rates* superiores..

Capítulo 6

6. Conclusões e Trabalho Futuro

Tendo em conta os objectivos referidos à partida para esta dissertação, estudaram-se alguns dos sistemas de vigilância inteligentes usados nos dias de hoje. Com base nesse estudo partiu-se para o estabelecimento dos principais objectivos, a definição de uma arquitectura para um sistema de vigilância distribuído e o desenvolvimento de um protótipo.

Neste capítulo final, é apresentado um resumo dos resultados obtidos, bem como propostas para futuros desenvolvimentos do sistema.

6.1. Satisfação dos Objectivos

O trabalho efectuado ao longo do período de desenvolvimento da dissertação, permitiu atingir os objectivos delineados.

Inicialmente foi feita uma análise da evolução dos sistemas de vigilância, percebendo-se o seu desenvolvimento ao longo da primeira, segunda e terceira gerações. Posteriormente foi efectuada uma análise crítica a alguns sistemas de terceira geração. Através desta análise concluiu-se que alguns sistemas possuem vulnerabilidades tais como: arquitecturas centralizadas ou apenas sistemas semi-distribuídos; dependência das condições meteorológicas; número reduzido de câmaras.

Após percepção das debilidades e vantagens de cada sistema, procurou-se conceber uma arquitectura que respeite os requisitos traçados, destacando-se a flexibilidade, robustez e escalabilidade. Com esta arquitectura modular flexível, o sistema pode adaptar-se a cenários de diferentes complexidades. Tal como os sistemas de vigilância de terceira geração, a presente arquitectura analisa informação proveniente de diversas fontes.

O desenvolvimento do protótipo foi baseado no projecto ZoneMinder. Os novos módulos integrados no ZoneMinder permitem usufruir de mais capacidade de decisão e detalhe, através de algoritmos de *tracking* e *segmentação*, podendo-se agora consultar no histórico o resultado do *tracking*, bem como consultar o número de pessoas detectadas numa câmara.

Os resultados apresentados apesar de uma limitação na capacidade de análise de frames, mostraram um funcionamento robusto da implementação ao nível da detecção do número de pessoas.

6.2. Trabalho Futuro

O protótipo desenvolvido permite a detecção de movimento e efectuar o *tracking* de pessoas. Contudo ainda é possível evoluir este protótipo de forma a se aproximar da arquitectura especificada.

A continuação do desenvolvimento do sistema passará por integrar um bloco de análise de áudio, possibilitando ao ZoneMinder analisar conjuntamente áudio e vídeo.

No que diz respeito à análise de vídeo, existe ainda espaço para progredir, no sentido de adicionar blocos de análise comportamental e *tracking multi-câmera*.

Tal como previsto na especificação, outra linha de desenvolvimento do protótipo, será o desenvolvimento de um componente que permita a interacção deste com sistemas externos.

Finalmente, uma etapa futura de grande interesse seria a validação do sistema desenvolvido em ambiente real.

Bibliografia

André, B. S.; (2004 Setembro). "Caracterização do desempenho de métodos de detecção de movimento aplicado a localização de pessoas através de visão computacional", *XV Congresso Brasileiro de Automática, Gramado, RS, Brasil.*

Black, J., Ellis, T., & Makris, D.; (2004). "A hierarchical database for visual surveillance applications", *In Proceedings of IEEE International Conference on Multimedia and Expo,* (pp. 1571-1574).

Collins R., Lipton A., & Kanade T.; (2000) "Introduction to the special section on video surveillance". *IEEE Trans. Pattern Analysis and Machine Intelligence*, (Vol. 22 No. 8 pp.745–746).

Cucchiara, R., Grana, C., Patri, A., Tardini, G., & Vezzani, R.; (2004) "Using computer vision techniques for dangerous situation detection in domotic applications". *Proc. IEE Workshop on Intelligent Distributed Surveillance Systems*, (pp. 1–5).

Detmold, H., van den Hengel, A., Dick A., Falkner K., Munro, D., & Morrison, R.; (2006)"Middleware for Video Surveillance Networks". *ACM International Conference Proceeding Series,* (Vol. 218, pp. 31-36).

Detmold, H., Dick, A., Falkner, K., Munro, D., Hengel, A., & Morrison. R.; (2006, Novembro). "Scalable Surveillance *Software* Architecture". *Proc. IEEE Conf. Advanced Video and Signal Based Surveillance*, (pp. 103-103).

Detmold, H., Van den Hengel, A., Dick A., Cichowski, A., Hill, R., Kocadag, E., Falkner, K. & Munro, D.; (2007, Setembro). "Topology Estimation for Thousand-Camera Surveillance Networks". *Proc. ACM/IEEE International Conference on Distributed Smart Cameras*, (pp. 195-202).

Dick A., & Brooks M. J.; (2004) A stochastic approach to *tracking* objects across multiple cameras. In *Proc. Australian Joint Conference on Artificial Intelligence*, (pp. 160–170, 2004).

Durães, D., Teixeira, L. F., & Corte-Real, L.; (2008, Julho) "Building modular surveillance systems based on multiple sources of information", *International Conference on Signal Processing and Multimedia Applications.*(Aceite)

Foresti, G.L.; (1999,Outubro). "Object recognition and *tracking* for remote video surveillance Circuits and Systems for Video Technology", *IEEE Transactions*, (7ºEd.Vol. 9, pp.1045 - 1062).

Franco, O., Giancarlo F. & Regazzoni C. S.; (2001, Setembro) "Recognition Driven Burst, Transmissions In Distributed Third Generation Surveillance Systems". (pp. 490-494).

Greenhill, D., Remagnino, P., & Jones, G.A.; (2002) "VIGILANT: contentquerying of video surveillance streams", *in Remagnino, P., Jones, G.A., Paragios, N., and Regazzoni, C.S. (Eds.): "Video-based Surveillance Systems"*, pp. 193–205

Hall D., Ribeiro P., Andrade E., Moreno P., Pesnel S., List T., Emonet R., Fisher R. B., Santos V. J., & Crowley J. L.(2005, Outubro). "Comparison of target detection algorithms using adaptative background models.", *In Proc. 2nd Joint IEEE Int. Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*.

Javed O., Rasheed Z., Alatas O., & Shah M.; (2003, Julho) "KNIGHTM: A real time surveillance system for multiple overlapping and non-overlapping cameras".

Javed, O. & Shah, M.; (2003) "KNIGHTM: A Multi-Camera Surveillance System". *ONDCP International Technology Symposium.*

Javed, O.; Shafique,K. & Shah, M.; (2007, Janeiro/Março) "Automated Surveillance in Realistic Scenarios". *IEEE MultiMedia*. (vol.14, no. 1, pp. 30-39).

Jianming, X.S., Yangsheng X., & Song J.; (2005) "Architecture and simulation of sensor network system for urban surveillance". *IEEE International Conference*, (pp.640-645).

Kumar, P., Mittal, A. & Kumar, P.; (2008) "Study of Robust and Intelligent Surveillance in Visible and Multimodal Framework".(vol. 32, no. 1, pp.63-77).

Lara, A.,C.; (2006, Junho) "*Segmentação* de Movimento para Aplicações de Vigilância Eletrônica" Departamento de Ciência da Computação, Instituto de Matemática e Estatística, Universidade de São Paulo.

MaKris, D., Ellis, T., & Black, J.; (2004, Junho) "Bridging the gaps between cameras". *Int. Conf. Multimedia and Expo.*

Oates, T., Schmill, M.D., & Cohen, P.R.; (2000) "A method for clustering the experiences of a mobile robot with human judgements'. *Proc. of the 17th National Conf. on Artificial Intelligence and Twelfth Conf. on Innovative Applications of Artificial Intelligence*, (pp. 846–851)

Patricio, M.A., Carbó, J., Pérez, O., Garcia, J. & Molina. J. M.; (2007, Janeiro) "Multi-Agent Framework in Visual Sensor Networks" *EURASIP Journal on Applied Signal Processing*, (pp- 226-226).

Pavlidis, I., Morellas, V., Tsiamyrtzis, P. & Harp, S.; (2001) "Urban surveillance systems: from the laboratory to the commercial world", *Proc. IEEE*, (Vol 89, (10), pp. 1478–1495).

Pozzobon, A., Sciutto, G., & Recagno, V.; (1998) "Security in ports: the user requirements for surveillance system". *Advanced Video-based Surveillance Systems*.

Rath, T.M., & Manmatha, R.; (2003) "Features for word spotting in historical manuscripts". *Proc. of the 7th Int. Conf. on Document Analysis and Recognition*, (pp. 512–527)

Shah, M., Javed, O. & Shafique, K.; (2007, Janeiro-Março) "Automated Visual Surveillance in Realistic Scenarios". *Multimedia, IEEE,* (vol. 14, no. 1, pp.30-39).

Stauffer C., & W. Grimson E. L.; (200 Agosto) "Learning patterns of activity using real-time tracking". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (Vol.22 no.8).

Sun J., Velastin S. A., & Lo B.; (2004) "A Distributed surveillance system to improve personal security in public transport"

Teixeira, L. F., Cardoso, J. S. & Corte-Real, L.; (2007, Setembro)." Object Segmentation Using Background Modelling and Cascaded Change Detection" (Vol.2 no. 5 pp.55-64)

Teixeira L. F., Carvalho P., Cardoso J. S. & Corte-Real. L.; (2008 Junho), **"**Automatic description of object appearances in a wide-area surveillance scenario". (Submetido)

Valera, M. & Velastin, S. A.; (2005). "Intelligent distributed surveillance systems: A review". *IEE Proceedings: Vision, Image and Signal Processing*, (pp: 152(2):192–204).

Valera, M. & Velastin, S. A.; (2008). "Middleware for distributed video surveillance" *IEEE Distributed Systems Online*, (Vol. 9, no.2).

Velastin, S.A., Boghossian, B.A., Lo, B.P.L., Sun, J. & Silva, V. M.A.; (2005, Janeiro)."PRISMATICA: toward ambient intelligence in public transport environments Systems, Man and Cybernetics". *IEEE Transactions*. (Ed. 1 ;Vol. 35) Part A, (pp. 164 – 182).

Zhao T. & Nevatia R.; (2004, Setembro) "*Tracking* Multiple Humans in Complex Situations". *IEEE Translations on pattern analysis and machine intelligence.* (Vol. 26, No. 9)

6.3. Referências Web

http://www.zoneminder.com/ (5/5/2008)

http://www.zoneminder.com/wiki/ (5/5/2008)

http://www.zoneminder.com/forums/ (6/5/2008)

http://www.zoneminder.com/wiki/index.php/Documentation. (7/5/2008)

http://www.debianpt.org/node/886 (7/5/2008)

http://utilidadelinux.blogspot.com/2007/06/segurana-patrimonial-com-o-zoneminder.html (7/5/2008)

http://www.guiado*hardware*.net/tutoriais/sistema-vigilancia-zoneminder/ (7/5/2008)

ANEXO A: BUILDING MODULAR SURVEILLANCE SYSTEMS BASED ON MULTIPLE SOURCES OF INFORMATION

BUILDING MODULAR SURVEILLANCE SYSTEMS BASED ON MULTIPLE SOURCES OF INFORMATION

Architecture and Requirements

Daniel Durães, Luís F. Teixeira, Luís Corte-Real

INESC Porto, Faculdade de Engenharia, Universidade do Porto

Campus da FEUP, Rua Dr. Roberto Frias, no. 378 4200-465 Porto, Portugal

ee03086@fe.up.pt, luis.f.teixeira@inescporto.pt, lreal@inescporto.pt

Keywords: Intelligent Surveillance Systems, Modular Architecture, Multi-sensor surveillance, Data Analysis.

Abstract: Intelligent surveillance is becoming increasingly important for the enhanced protection of facilities such as airports and power stations from various types of threats. We propose a surveillance system architecture based on multiple sources of information to apply on large scale surveillance networks. The main contribution of this paper is the definition of the requirements for a flexible and scalable architecture that supports intelligent surveillance using, alongside video, different sources of information, such as audio or other sensors.

1 INTRODUCTION

Visual surveillance systems are widely applied in transport scenarios, such as airports, railways, underground, and motorways as well as other public spaces, such as banks and shopping malls. Typically these systems are manually handled by a human operator and do not rely on content-based automation processes. The captured content is stored for a limited period of time, due to storage limitations. Also, searching for a specific event can be a very time consuming task. Recent advances in this area are opening new possibilities for the next generation surveillance systems (Valera and Velastin, 2005). The main focus of research is on improving image processing tasks by generating more accurate and robust algorithms for object detection and recognition, tracking and human activity recognition.

A new generation of systems for surveillance are also starting to be commercialised. The common processing tasks that commercial systems perform are intrusion, motion detection and packages detection. Typical examples of commercial surveillance systems are DETEC [1] and Gotcha [2]. They are usually based on what is commonly called motion detectors, with the option of digital storage of the detected events. These events are usually triggered by objects appearing in the scene. Another example of a commercial system intended for outdoor applications is DETER (Pavlidis et al., 2001), which reports unusual movement patterns of pedestrians and vehicles in outdoor environments such as car parks. A broader goal is defined by the PRISMATICA project (Lo et al., 2003). The developed system is a wide-area multisensor distributed system, receiving inputs from CCTV, local wireless camera networks, smart cards, and audio sensors. Also, the project ADVISOR (Siebel and Maybank, 2004) aims to assist human operators by automatically selecting, recording, and annotating images containing events of interest. ADVISOR interprets shapes and movements in scenes being viewed by the CCTV to build up a picture of the behavior of people in the scene.

In summary, intelligent surveillance systems are not restricted to cameras but can also contain different types of information sources, to better interpret the danger. Monitoring surveillance networks by human inspection is expensive and ineffective. Consequently, surveillance users are choosing software for automated video surveillance. The architecture presented, proposes a solution to integrate specific surveillance algorithms, acquire information from various sources and interact with external systems. The solution presented, specifies architecture organized in a hierarchical structure and divided into

[1] http://www.detec.no

[2] http://www.gotchanow.com

different modules, including both support for communication and for computation.

2 SYSTEM REQUIREMENTS

Our objective is to devise a software architecture for surveillance systems composed of a sensor network of different sources. The architecture should allow adding new and improved surveillance techniques while the network continues operating. The requirements are divided in non-functional, functional and hardware categories. The first two categories of requirements are based on (Detmold et al., 2006) and (Valera and Velastin, 2008).

2.1 Non-Functional Requirements

The non-functional requirements for video surveillance networks include scalability, availability, evolvability, integration, security and manageability.

- *Scalability*: We should consider different scopes for scalability. Processing the data generated in a large-scale surveillance system in a centralized architecture is unfeasible and a scalable distributed processing is required. Also, storing the relevant data, even if a small fraction of all captured data, requires a scalable storage distributed system. Finally, in some cases a scalable network needs to be considered, mainly if the number of sources of information are expected to increase.
- *Availability*: A larger number of components increases the probability that some component will eventually fail. However, the architecture must support acceptable levels of operation. A configuration of redundant systems should be considered if high level of availability is required.
- *Evolvability*: Within some limits, the surveillance network should be able to accommodate changes, including alterations to the hardware and changes to the software.
- *Integration*: Nowadays surveillance systems usually operate in independently of other systems. However, there is a growing need to integrate different systems, especially for intelligent management of buildings. It is therefore required a perspective of integration with external systems.
- *Security*: The integration with external systems however intensifies the need for security. A critical requirement is that the system should be robust to attacks with malicious intents. The consequences of such attacks include compromising confidential information or shutting down the system through denial of service attacks.

2.2 Functional Requirements

The functional requirements of intelligent surveillance systems include modules that perform: (1) signal processing, including audio and visual processing, (2) data aggregation and higher-level semantic analysis, (3) command, control and inspection of all network elements, and (4) storage with browsing and forensic analysis capabilities.

We first consider a low-level *signal processing* module that receives data from sensors and generates a sparser representation of data when compared to the raw data. For example, for the visual sensors, i.e. cameras, the signal processing module includes: *object detection*, *object classification* according to shape, color, and other properties, and *object tracking* along time within individual cameras views. In this case, the signal processing module accepts images from individual cameras as input, and produces as output a compact representation of the content – for example, the object trajectories. These modules are closely related to each sensor and can be seen as a layer just above the sensors. Depending on the hardware capabilities of the sensors, the modules' implementation could be embedded in the sensors.

The data processed for individual sensors is collected by a *data aggregation* module that relies on the multiple sources of data in the network to get a more complete representation. Using the same example, for visual sensors the data aggregation module includes: *multicamera tracking of objects* across time and multiple cameras views. Also, a *higher-level analysis* is performed to extract semantic knowledge, which usually requires a priori information and training. One objective could be *behavioral analysis*, or in other words, to recognize and understand the activities of the tracked objects. The output of these modules is typically auxiliary data (or metadata) that is stored alongside the raw data or events to alert the human operator.

The events triggered by the system warn about situations that eventually require close attention by the operators. When needed, the operators use *command control, and inspection* functionalities to interact with system – e.g. examine the status of an element (e.g. a sensor) in the network and take corrective actions. Other examples of interactions include remote *sensor control* of individual sensors in real-time, *target following* by selecting an object to follow and making a report, and *external system control* namely, controlling the elevators system or locking the doors.

Finally, *storage with browsing and forensic analysis* capabilities allow an operator to efficiently find a given event or object. Browsing or querying the historical archive, the operator can look for detected events and know the details of possible threats. The auxiliary data mentioned previously is used to aid browsing and provide forensic analysis. Exploring large amounts of data without such capabilities can be extremely time consuming and inefficient.

2.3 Hardware Requirements

This architecture proposes the sharing of data though different interfaces. The interfaces depend on the needs of the system but typically the whole hardware is connected by a local area network (LAN).

The system is composed of computers or smart sensors (mostly cameras) connected by the LAN. A human-computer interface and a storage farm are also plugged in this system. The type of sensors that are possible connect besides cameras and microphones are smoke sensors, biometric sensors among others.

The storage module needs to allow 24 hours/day of data. Since only relevant data is stored, the amount of data produced in a day may vary. If we use MPEG-4 technology at 25 fps and 160 Kbit/s, in 3 days the total amount of raw data for a camera is 40 GB. However only a much smaller fraction of this needs to be stored ($\sim 5\%$).

A distributed system implies an efficient use of the available bandwidth that satisfies quality of service (QoS).

3 ARCHITECTURE

The concept of the architecture is based on a hierarchical relation between elements, respecting functional and hardware requirements and allowing interaction with external systems. Figure 1 shows a physical view of the surveillance system architecture using the standard UML model representation.

The main modules in the architecture are *control and command*, *database centre* and *monitoring*. All these modules contain a component called *communications manager*. Its goal is to manage the communications between different modules.

3.1 Control and Command

With the control and command module, human operators can visualize the outputs of the system and eventually take an action. The interaction between operators and the system is done through a graphical user interface (GUI) is possible interact with system.

A possible implementation solution is to use a web-based GUI, implemented in PHP or Java (server-side) and HTML with Javascript (client-side).

3.2 Database Centre

The database centre module is a physically independent module, where storage all data produced by system. This architecture is both flexible, because the database centre module is independent of other modules, and scalable, because many database centres may operate together or separately. The redundancy provided by multiple database centres is also essential to maintain high availability of operation.

In (Black et al., 2004), the authors present a solution that propose a creation of a database with four layers, ordered hierarchically, supporting high and low level queries. The data is stored selectively by the different layers. The system provides a mechanism to generate video content summaries of objects detected by the system across the entire surveillance region in terms of the semantic scene model. The layers of abstraction are: (1) raw data layer, (2) object motion layer, (3) semantic description layer, and (4) metadata layer. At the lowest level the system user can manually browse a stored video to observe some object activity recorded by the system. At the highest level, queries could be executed to automatically find that same object activity.

3.3 Monitoring

This is the system module closer to the hardware. Monitoring is performed independently for each surveillance sensor and afterwards aggregated by a multimodal data analysis system. The monitoring module contains the following components:

- *Signal processing, including visual and audio analysis*: Processes the data captured by the distributed network of sensors. For the visual sensors it typically includes object detection, classification and tracking. A similar flow is done for audio processing, including audio source separation and classification to detects dangerous situations, such as screams or explosions. If an array of microphones is available, localization and tracking of audio sources is also possible. Each of the components (visual analysis and audio analysis) is associated to a single sensor.
- *Analysis of data from other sources*: Analyses the information obtained by other sources, such

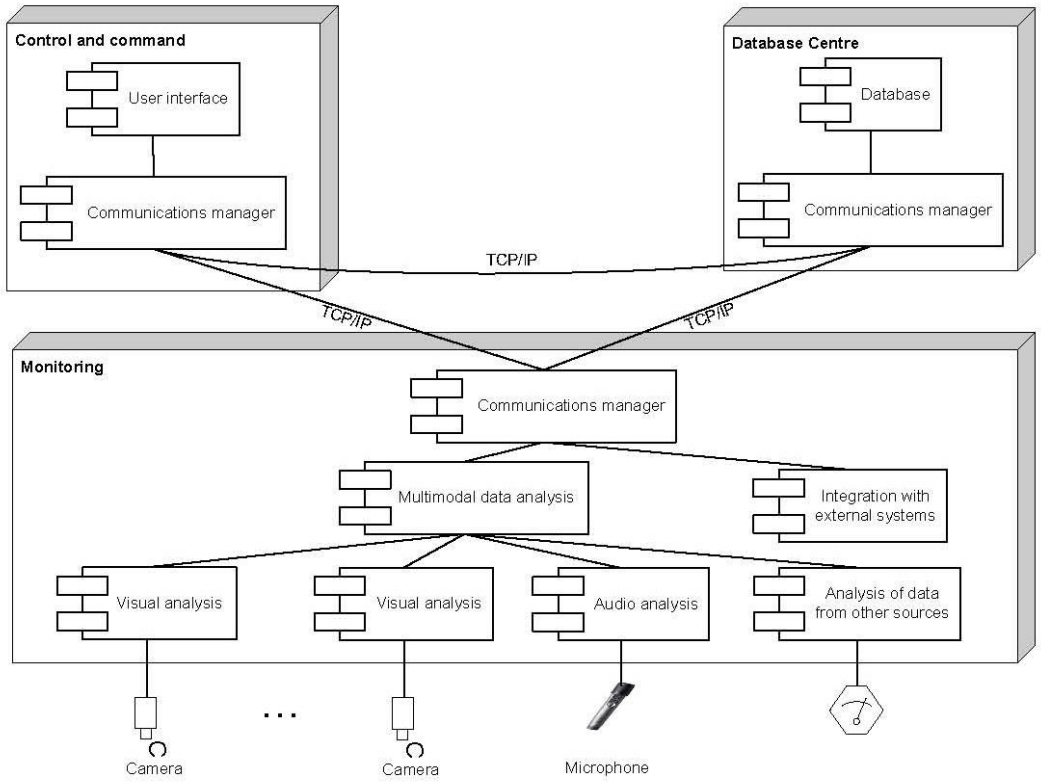


Figure 1: Component diagram of the system architecture.

as emergency door sensors, fire detection alarms, etc.

- *Multimodal data analysis*: Receives data from the signal processing and the analysis of data from other sources components. Performs data aggregation and fusion, such as multicamera tracking and behavioral analysis. The processed data and generated events are sent to the storage module. If a give event is configured to generate an alarm, the human operator is notified by the control and command module.
- *Integration with external systems*: Interfaces with external systems and executes the orders given by human operator, for example building access.

With this architecture based in modules and components it is possible to capture different types of information and analyse them to produce better results when compared to the results obtained by a separate analysis of sensor information. The ultimate goal is to minimize the errors generated by the system.

The components performing data analysis (including visual, audio, other sources and multimodal analysis) are typically computationally intensive and require a low-level implementation in C or C++. The integration of individual components can be implemented using higher-level programming languages, such as Perl and Python.

4 IMPLEMENTATION

The system is currently being developed and not all specified modules are fully functioning. In this section we give an overview of the current status of implementation.

4.1 Algorithms

Object detection is done using cascade algorithm (Teixeira et al., 2007) essentially based on mixture of Gaussians for background modelling (Lee, 2005). Taking in consideration that background changes are caused by phenomena of different nature, a cascaded evaluation of typical dynamic elements is performed. These elements include acquisition noise, illumination variation and slow or repetitive structural changes. The latter type of changes is classified using the methods that estimate a p.d.f. to model the background. Also, a statistical test and a simple collinearity test are used to classify changes originated by noise and illumination changes, respectively. Some results are shown in Figure 2.

Figure 2: Segmentation results (second row) using the cascaded change detection algorithm and tracking results (third row) using the Kalman filtering with appearance constraints.

Tracking of objects is done using an implementation based on (Zhao and Nevatia, 2004). This tracking method relies on Kalman filtering for hypothesis tracking, and on the objects appearance constrained by its shape.

Tracks from different objects are matched to find multiple appearances of an object as in Figure 3. This is accomplished with a combined representation scheme and incremental object model to find matches between visual object tracks (Teixeira and Corte-Real, 2008). The description scheme relies on SIFT local descriptors and a text-like bag-of-words representation. Results show that this object representation scheme can be used to aid tracking of generic objects in visual surveillance systems, since it can discriminate a large quantity of different visual objects very well, and can be adapted to reflect object changes. It also presented a good resilience to incorrect segmentations when extracting the visual objects from complex scenes. The object model is updated through incremental learning, avoiding excessive data storage while maintaining performance and allowing new objects to be learnt by the system.

Other analysis algorithms, namely audio content analysis, will be added in the future.

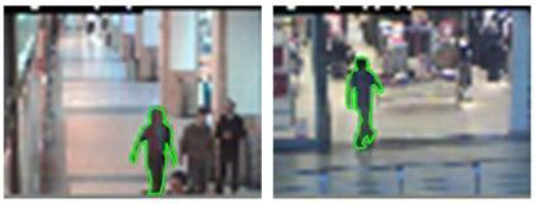


Figure 3: Same visual object captured at different instants, by different cameras.

4.2 Prototype

The current prototype of the proposed architecture is based on ZoneMinder [3]. This choice is due to the fact that ZoneMinder is an open-source surveillance framework that includes functionalities of typical surveillance systems – supports capture, analysis, recording, and monitoring of multiple video cameras. Given a state-of-the-art surveillance framework, which closely follows the architecture defined in Section 3, our goal is to instantiate the requirements stated in Section 2.

The ZoneMinder framework comprises different components written in C++ (core), PHP (GUI) and Perl (scripts). A motion analysis module detects activity in the captured areas, allowing selective recording defined by the user. The user can also define alarms, such as intrusion in protected areas, which are stored in a MySQL database as events. These events are associated with the corresponding video for future visualization. These management and visualization functions are accomplished through a web-based GUI.

The analysis functionalities provided by typical surveillance systems, such as ZoneMinder, are limited to activity detection based on motion analysis and alarm generation based on the definition of protected areas. In the prototype we extended these functionalities with advanced detection such as object segmentation, tracking and matching (see Section 4.1). In summary, and considering the database model proposed in (Black et al., 2004), our prototype populates automatically the semantic layer, in contrast to the typical systems that generate information for the image and motion layers.

5 CONCLUSIONS

This paper introduces a modular surveillance systems architecture supporting the requirements of au-

[3] http://www.zoneminder.com

tomated video surveillance networks, which are an increasingly important tool for preventing and countering security threats. The architecture can be adapted to different scenarios. Also, unlike typical surveillance systems, multiple sources of information are considered. A prototype based on open-source framework for visual surveillance was implemented. The prototype includes algorithms for the segmentation and tracking of visual objects. Another module that detects multiple appearances of the detected visual objects was also devised and integrated in the system. Future work includes integrating of audio analysis modules and the support for evolvability and availability.

REFERENCES

Black, J., Ellis, T., and Makris, D. (2004). A hierarchical database for visual surveillance applications. In *Proceedings of IEEE International Conference on Multimedia and Expo*, pages 1571–1574.

Detmold, H., Dick, A., Falkner, K., Munro, D. S., van den Hengel, A., and Morrison, R. (2006). Scalable surveillance software architecture. In *Proceedings of IEEE International Conference on Video and Signal Based Surveillance*, pages 103–106.

Lee, D.-S. (2005). Effective Gaussian mixture learning for video background subtraction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(5):827–832.

Lo, B., Sun, J., and Velastin, S. (2003). Fusing visual and audio information in a distributed intelligent surveillance system for public transport systems. *Acta Automatica Sinica*, 29(3):393–407.

Pavlidis, I., Morellas, V., Tsiamyrtzis, P., and Harp, S. (2001). Urban surveillance systems: from the laboratory to the commercial world. *Proceedings of the IEEE*, 89(10):1478–1497.

Siebel, N. T. and Maybank, S. J. (2004). The ADVISOR visual surveillance system. In *Proceedings of the ECCV Workshop on Applications of Computer Vision*, pages 103–111.

Teixeira, L. F., Cardoso, J. S., and Corte-Real, L. (2007). Object segmentation using background modelling and cascaded change detection. *Journal of Multimedia*, 2(5):55–64.

Teixeira, L. F. and Corte-Real, L. (2008). Video object matching across multiple independent views using local descriptors and adaptive learning. *Pattern Recognition Letters*. (in press).

Valera, M. and Velastin, S. A. (2005). Intelligent distributed surveillance systems: A review. *IEE Proceedings: Vision, Image and Signal Processing*, 152(2):192–204.

Valera, M. and Velastin, S. A. (2008). Middleware for distributed video surveillance. *IEEE Distributed Systems Online*, 9(2).

Zhao, T. and Nevatia, R. (2004). Tracking multiple humans in complex situations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1208–1221.