FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Title of the Dissertation

Name of the Author

WORKING VERSION

Mestrado Integrado em Engenharia Informática e Computação

Supervisor: Name of the Supervisor

June 20, 2017

Title of the Dissertation

Name of the Author

Mestrado Integrado em Engenharia Informática e Computação

June 20, 2017

Abstract

Here goes the abstract written in English.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Sed vehicula lorem commodo dui. Fusce mollis feugiat elit. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec eu quam. Aenean consectetuer odio quis nisi. Fusce molestie metus sed neque. Praesent nulla. Donec quis urna. Pellentesque hendrerit vulputate nunc. Donec id eros et leo ullamcorper placerat. Curabitur aliquam tellus et diam.

Ut tortor. Morbi eget elit. Maecenas nec risus. Sed ultricies. Sed scelerisque libero faucibus sem. Nullam molestie leo quis tellus. Donec ipsum. Nulla lobortis purus pharetra turpis. Nulla laoreet, arcu nec hendrerit vulputate, tortor elit eleifend turpis, et aliquam leo metus in dolor. Praesent sed nulla. Mauris ac augue. Cras ac orci. Etiam sed urna eget nulla sodales venenatis. Donec faucibus ante eget dui. Nam magna. Suspendisse sollicitudin est et mi.

Fusce sed ipsum vel velit imperdiet dictum. Sed nisi purus, dapibus ut, iaculis ac, placerat id, purus. Integer aliquet elementum libero. Phasellus facilisis leo eget elit. Nullam nisi magna, ornare at, aliquet et, porta id, odio. Sed volutpat tellus consectetuer ligula. Phasellus turpis augue, malesuada et, placerat fringilla, ornare nec, eros. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Vivamus ornare quam nec sem mattis vulputate. Nullam porta, diam nec porta mollis, orci leo condimentum sapien, quis venenatis mi dolor a metus. Nullam mollis. Aenean metus massa, pellentesque sit amet, sagittis eget, tincidunt in, arcu. Vestibulum porta laoreet tortor. Nullam mollis elit nec justo. In nulla ligula, pellentesque sit amet, consequat sed, faucibus id, velit. Fusce purus. Quisque sagittis urna at quam. Ut eu lacus. Maecenas tortor nibh, ultricies nec, vestibulum varius, egestas id, sapien.

Phasellus ullamcorper justo id risus. Nunc in leo. Mauris auctor lectus vitae est lacinia egestas. Nulla faucibus erat sit amet lectus varius semper. Praesent ultrices vehicula orci. Nam at metus. Aenean eget lorem nec purus feugiat molestie. Phasellus fringilla nulla ac risus. Aliquam elementum aliquam velit. Aenean nunc odio, lobortis id, dictum et, rutrum ac, ipsum.

Ut tortor. Morbi eget elit. Maecenas nec risus. Sed ultricies. Sed scelerisque libero faucibus sem. Nullam molestie leo quis tellus. Donec ipsum. Nulla lobortis purus pharetra turpis. Nulla laoreet, arcu nec hendrerit vulputate, tortor elit eleifend turpis, et aliquam leo metus in dolor. Praesent sed nulla. Mauris ac augue. Cras ac orci. Etiam sed urna eget nulla sodales venenatis. Donec faucibus ante eget dui. Nam magna. Suspendisse sollicitudin est et mi.

Phasellus ullamcorper justo id risus. Nunc in leo. Mauris auctor lectus vitae est lacinia egestas. Nulla faucibus erat sit amet lectus varius semper. Praesent ultrices vehicula orci. Nam at metus. Aenean eget lorem nec purus feugiat molestie. Phasellus fringilla nulla ac risus. Aliquam elementum aliquam velit. Aenean nunc odio, lobortis id, dictum et, rutrum ac, ipsum.

Ut tortor. Morbi eget elit. Maecenas nec risus. Sed ultricies. Sed scelerisque libero faucibus sem. Nullam molestie leo quis tellus. Donec ipsum.

Resumo

O Resumo fornece ao leitor um sumário do conteúdo da dissertação. Deverá ser breve mas conter detalhe suficiente e, uma vez que é a porta de entrada para a dissertação, deverá dar ao leitor uma boa impressão inicial.

Este texto inicial da dissertação é escrito no fim e resume numa página, sem referências externas, o tema e o contexto do trabalho, a motivação e os objectivos, as metodologias e técnicas empregues, os principais resultados alcançados e as conclusões.

Este documento ilustra o formato a usar em dissertações na Faculdade de Engenharia da Universidade do Porto. São dados exemplos de margens, cabeçalhos, títulos, paginação, estilos de índices, etc. São ainda dados exemplos de formatação de citações, figuras e tabelas, equações, referências cruzadas, lista de referências e índices. É usado texto descartável, *Loren Ipsum*, para preencher a dissertação por forma a ilustrar os formatos.

Seguem-se umas notas breves mas muito importantes sobre a versão provisória e a versão final do documento. A versão provisória, depois de verificada pelo orientador e de corrigida em contexto pelo autor, deve ser publicada na página pessoal de cada estudante/dissertação, juntamente com os dois resumos, em português e em inglês; deve manter a marca da água, assim como a numeração de linhas conforme aqui se demonstra.

A versão definitiva, a produzir somente após a defesa, em versão impressa (dois exemplares com capas próprias FEUP) e em versão eletrónica (6 CDs com "rodela" própria FEUP), deve ser limpa da marca de água e da numeração de linhas e deve conter a identificação, na primeira página, dos elementos do júri respetivo. Deve ainda, se for o caso, ser corrigida de acordo com as instruções recebidas dos elementos júri.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Sed vehicula lorem commodo dui. Fusce mollis feugiat elit. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Donec eu quam. Aenean consectetuer odio quis nisi. Fusce molestie metus sed neque. Praesent nulla. Donec quis urna. Pellentesque hendrerit vulputate nunc. Donec id eros et leo ullamcorper placerat. Curabitur aliquam tellus et diam.

Ut tortor. Morbi eget elit. Maecenas nec risus. Sed ultricies. Sed scelerisque libero faucibus sem. Nullam molestie leo quis tellus. Donec ipsum. Nulla lobortis purus pharetra turpis. Nulla laoreet, arcu nec hendrerit vulputate, tortor elit eleifend turpis, et aliquam leo metus in dolor. Praesent sed nulla. Mauris ac augue. Cras ac orci. Etiam sed urna eget nulla sodales venenatis. Donec faucibus ante eget dui. Nam magna. Suspendisse sollicitudin est et mi.

Phasellus ullamcorper justo id risus. Nunc in leo. Mauris auctor lectus vitae est lacinia egestas. Nulla faucibus erat sit amet lectus varius semper. Praesent ultrices vehicula orci. Nam at metus. Aenean eget lorem nec purus feugiat molestie. Phasellus fringilla nulla ac risus. Aliquam elementum aliquam velit. Aenean nunc odio, lobortis id, dictum et, rutrum ac, ipsum.

Ut tortor. Morbi eget elit. Maecenas nec risus. Sed ultricies. Sed scelerisque libero faucibus sem. Nullam molestie leo quis tellus. Donec ipsum.

Acknowledgements

Aliquam id dui. Nulla facilisi. Nullam ligula nunc, viverra a, iaculis at, faucibus quis, sapien. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Curabitur magna ligula, ornare luctus, aliquam non, aliquet at, tortor. Donec iaculis nulla sed eros. Sed felis. Nam lobortis libero. Pellentesque odio. Suspendisse potenti. Morbi imperdiet rhoncus magna. Morbi vestibulum interdum turpis. Pellentesque varius. Morbi nulla urna, euismod in, molestie ac, placerat in, orci.

Ut convallis. Suspendisse luctus pharetra sem. Sed sit amet mi in diam luctus suscipit. Nulla facilisi. Integer commodo, turpis et semper auctor, nisl ligula vestibulum erat, sed tempor lacus nibh at turpis. Quisque vestibulum pulvinar justo. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Nam sed tellus vel tortor hendrerit pulvinar. Phasellus eleifend, augue at mattis tincidunt, lorem lorem sodales arcu, id volutpat risus est id neque. Phasellus egestas ante. Nam porttitor justo sit amet urna. Suspendisse ligula nunc, mollis ac, elementum non, venenatis ut, mauris. Mauris augue risus, tempus scelerisque, rutrum quis, hendrerit at, nunc. Nulla posuere porta orci. Nulla dui.

Fusce gravida placerat sem. Aenean ipsum diam, pharetra vitae, ornare et, semper sit amet, nibh. Nam id tellus. Etiam ultrices. Praesent gravida. Aliquam nec sapien. Morbi sagittis vulputate dolor. Donec sapien lorem, laoreet egestas, pellentesque euismod, porta at, sapien. Integer vitae lacus id dui convallis blandit. Mauris non sem. Integer in velit eget lorem scelerisque vehicula. Etiam tincidunt turpis ac nunc. Pellentesque a justo. Mauris faucibus quam id eros. Cras pharetra. Fusce rutrum vulputate lorem. Cras pretium magna in nisl. Integer ornare dui non pede.

The Name of the Author

"You should be glad that bridge fell down.
I was planning to build thirteen more to that same design"

Isambard Kingdom Brunel

Contents

List of Figures

LIST OF FIGURES

List of Tables

LIST OF TABLES

Abbreviations

ADT	Abstract Data Type
ANDF	Architecture-Neutral Distribution Format
API	Application Programming Interface
CAD	Computer-Aided Design
CASE	Computer-Aided Software Engineering
CORBA	Common Object Request Broker Architecture
UNCOL	UNiversal COmpiler-oriented Language
Loren	Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Sed vehicula lorem commodo dui
WWW	*World Wide Web*

Todo list

ABBREVIATIONS

Chapter 1

Introdução

O primeiro capítulo da dissertação deve servir para apresentar o enquadramento e a motivação do trabalho e para identificar e definir os problemas que a dissertação aborda. Deve resumir as metodologias utilizadas no trabalho e termina apresentando um breve resumo de cada um dos capítulos posteriores.

1.1 Contexto/Enquadramento

Esta secção descreve a área em que o trabalho se insere, podendo referir um eventual projeto de que faz parte e apresentar uma breve descrição da empresa onde o trabalho decorreu.

1.2 Projeto

Na continuação da secção anterior, e apenas no caso de ser um Projeto e não uma Dissertação, esta secção apresenta resumidamente o projeto.

1.3 Motivation and Goals

Apresenta a motivação e enumera os objetivos do trabalho terminando com um resumo das metodologias para a prossecução dos objetivos.

1.4 Estrutura da Dissertação

Para além da introdução, esta dissertação contém mais x capítulos. No capítulo 2, é descrito o estado da arte e são apresentados trabalhos relacionados. No capítulo 3, ipsum dolor sit amet, consectetuer adipiscing elit. No capítulo 4 praesent sit amet sem. No capítulo 5 posuere, ante non tristique consectetuer, dui elit scelerisque augue, eu vehicula nibh nisi ac est.

Chapter 2

Revisão Bibliográfica

Neste capítulo é descrito o estado da arte e são apresentados trabalhos relacionados para mostrar o que existe no mesmo domínio e quais os problemas em aberto. Deve deixar claro que existe uma oportunidade de desenvolvimento que cobre alguma falha concreta . O capítulo deve também efetuar uma revisão tecnológica às principais ferramentas utilizáveis no âmbito do projeto, justificando futuras escolhas.

Resenha utilizaçao computer vision em trafego e em transportes Concluir com os desafios da armis (drive, quere aplicaro computer vision para automatizar determinados aspetos) Sequência de funcionalidades que respondem ao desafio da Armis(segmentaçao,deteçao,...) Uma secção para cada um

Tese Gustavo Lira / Pedro Loureiro

2.1 History of Computer Vision

Computer vision appeared as an area of investigation around the mid 60's at the MIT by Professor Larry Roberts, whose PhD. thesis focused on methods to extract 3D information from a 2D image [Hua96] in order to reconstruct entire scenes from the geometry gathered. This area is now considered by the ACM as a branch of artificial intelligence according to the 2012 ACM Computing Classification System [ACM12]. From this point onward scientists began applying the techniques developed in multiple fields.

The use of computer vision in manufacturing industry was among the first to be explored, as there were already multiple robots working in the assembly lines of the factories that needed to be improved, as noted by Stout in 1980 [Sto80]. These robots were becoming outdated due to the lack of interaction with the environment. This meant that for a robot to work with a part in an assembly line it needed to be placed in a pre-determined position, and any deviation could mean that the line needed to be stopped. Michael Baird relates in [LB78] a computer vision system developed to inspect circuit chips rotation deviation in a welding base, indicating to the welder that the chip needed to be adjusted. Computer vision was also used as a tool to automatically analyse weather satellite images [BT73] and multidimensional medical images [Aya98], with applications being developed in these areas being now common in our daily life.

2.1.1 Computer Vision in Intelligent Transport Systems

Regarding intelligent transport systems (ITSs), the use of computer vision to aid in the analysis of traffic has been increasing in the last years due to the decreasing costs of hardware, both cameras, storage and processing power, as well as the growing knowledge to extract useful data from the video gathered. In contrast to the high installation and upkeep costs of other traffic control tools such as Inductive Loop Detectors and Microwave Vehicle Detectors, applying computer vision to handle these tasks is a profitable option for entities in charge of the analysis of this data.

One of the first works applying computer vision to analyse traffic was published in 1984 [DW84] and detected and measured movement in a sequence of frames. Since then, multiple fields of study have been created, not only for the measuring of traffic, as explored in [LKR+16], [BVO11] and [HK12] where the authors evaluate methods to analyse traffic in urban environments, but also for the analysis of the environment around a self-driven vehicle, reading traffic signs using multiple approaches using convolutional neural networks [SS11] or using bag-of-visual-words [SLZ16], or to automatize the parking process as discussed in [HBJ+16]. Recently some studies have surfaced where the authors discuss the analysis of passenger numbers and behaviour inside vehicles, in order to enforce traffic laws [LBT17].

For our application however we need to focus on the aspect of traffic analysis, and in order to understand what traffic analysis software needs to accomplish, it is convenient to study what composes a typical traffic scene. Usually the scenes are viewed from a top-down perspective that places the cars against the road as background, as seen in 2.1. The vehicles have a roughly regular rectangular shape when viewed from the top, with little variation when the camera is rotated, however their textures vary heavily, making it difficult for a detector to work based on the vehicles image representation [BP98]. However, depending on the camera position there can be object occlusion by other objects or scene components, which makes it difficult to detect and track vehicles relying solely on subtraction based segmentation techniques.

The scenes also have varying light according to the time of the day, and proposed solutions need to adapt to these changes as fast as possible, otherwise data might be lost. Other weather conditions can also interfere with the analysis, such as fog and rain, and need to be addressed when designing solutions.

2.1.2 Our Challenge

The "Analytics Server" module of the "Video Server" project had a list of requirements that needed

Acabar

to be

Detetar eventos relevantes Contra mao Objetos caidos

Contar Veiculos

Figure 2.1: Typical High-Way Traffic Scene

Apenas para camaras fixas

2.2 Foreground Segmentation

Image segmentation is the process through which an image is separated into its different regions according to the desired output, usually objects of interest. These regions are composed by pixels that have a common characteristic [SS01] dependant on the desired result. There are several ways one can accomplish an adequate segmentation of an image, and the most relevant ones to our purpose are described below.

2.2.1 Background Subtraction

Background Subtraction is a segmentation technique based on the analysis of the difference of consecutive frames and use that information to create a background model, a representation of what the image looks like without any moving objects present. Given the nature of the process, cameras must be static and although some research has been made to overcome this issue [LCC12] [SJK09] [ZY14], this is beyond the scope of the project, due to the requirements given.

There are however a number of different ways to implement the background subtraction developed across the years with increasing segmentation accuracy and computational performance. Piccardi presented an overview of the most relevant ones in [Pic04], aiming to present each method's strengths and weaknesses. From this work we can present a list of the algorithms considered for implementation in the project.

2.2.1.1 Frame Differencing

This is the simplest approach to the problem as it only considers two frames at a time, making it only work in certain scenarios where the speed of the objects and frame rate of the camera allow it. In this model a pixel is considered as foreground if it satisfies the equation presented in 2.1, and since the only parametrizable value is the Threshold, the whole segmentation process is very sensitive to any changes in this value.

$$|frame_i - frame_{i-1}| >= Threshold \tag{2.1}$$

2.2.1.2 History Based Background Model

In order to make this segmentation method more robust, in the early 2000's Velastin [LV01] and Cucchiara [CGPP03] proposed a mathematical model to take into consideration past frames when calculating the Background Model of the scene. The most simple version of the improved algorithm used a running average of the past frames as seen in 2.2. This eliminates the need to store the frames in memory as the new average can be calculated using the previous results.

$$BackgroundModel_{i+1} = \alpha * Frame_i + (1 - \alpha) * BackgroundModel_i \tag{2.2}$$

Instead of the running average one can use the median value of the last n frames, improving the method reaction to outlying values. In both however, the history can be just the n frames or a weighted average where recent frames have more weight. Both these methods cannot cope with intermittent changes on the background, such as moving leaves against a building facade, and to solve this problem a new approach was proposed, the Mixture of Gaussians which models each pixel according to a mixture of Gaussian distributions, usually 3 to 5, but in later research [Ziv04] a method was developed to calculate the number of distributions needed on a per pixel basis.

2.2.2 Object Based

Object Based segmentation relies on the a priori knowledge of the geometry of the objects we want to extract from the image. It is mainly used in medical applications such as in [SMG+93] where it is used to extract a model of the brain surface from MRI scans

Acabar

2.3 Image Treatment

This section will review some of the proposed methods to improve image quality before or during processing by removing noise, extracting unwanted features or enhancing relevant features for the processes involved.

2.3.1 Blur

In order to blur an image one needs to convolve it using a special kind of filter. The process of convolution in image processing consists in the application of a filter to every pixel of an image, returning a new image where each pixel is the result of the function to the pixel in the same position in the previous image. As we can see in 2.2 the application of a 3*3 mask, where all 9 values have the value 1/9, results in a image where each pixel value equals the mean of itself and surrounding neighbours in the original image.

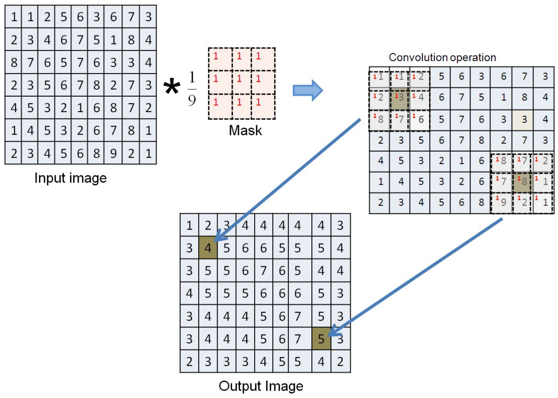


Figure 2.2: Convolution Example (src: http://cse19-iiith.vlabs.ac.in/theory.php?exp=neigh)

Verificar Imagem

While this is the simplest blur processing we can apply to an image, there are more complex ones that yield better results. The one most often used in image processing is the Gaussian Blur, which uses a square filter with values calculated with the equation in 2.3 from [SS01].

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{2.3}$$

This equation returns a filter consisting of concentric circles that results in a smooth blurring of the images due to the application of the same power of blurring from equidistant pixels. The function can be tuned to the needs of the process through the σ value, returning a sharper or softer blur filter as seen in 2.3.

This kind of filters have multiple uses in computer vision

2.3.2 Morphology Operators

2.3.3 Edge Detection

2.3.4 Shadow Removal

Rain Removal Shadow Removal

2.4 Secção Exemplo

Scalable Vector Graphics é uma linguagem em formato XML que descreve gráficos de duas dimensões. Este formato padronizado pela W3C (*World Wide Web Consortium*) é livre de patentes

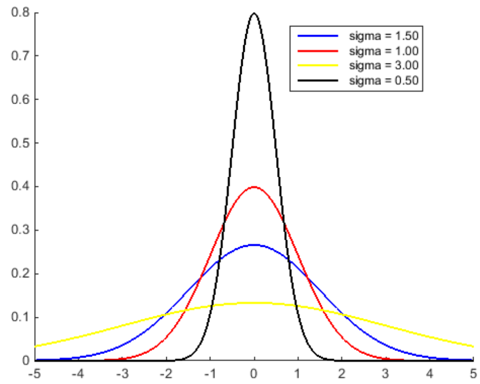


Figure 2.3: Gaussian Function Variation with Sigma Value

ou direitos de autor e está totalmente documentado, à semelhança de outros W3C standards citekn:svgdoc.

Sendo uma linguagem XML, o *SVG* herda uma série de vantagens: a possibilidade de transformar *SVG* usando técnicas como XSLT, de embeber *SVG* em qualquer documento XML usando *namespaces* ou até de estilizar *SVG* recorrendo a CSS (*Cascade Style Sheets*). De uma forma geral, pode dizer-se que *SVG*s interagem bem com as atuais tecnologias ligadas ao XML e à Web, tal como referido em citekn:svgibm,kn:svgw3c.

2.5 Resumo ou Conclusões

No final do capítulo deverá ser apresentado um resumo com as principais conclusões que se podem tirar.

Vivamus non nunc nec risus tempor varius. Quisque bibendum mi at dolor. Aliquam consectetuer condimentum risus. Aliquam luctus pulvinar sem. Duis aliquam, urna et vulputate tristique, dui elit aliquet nibh, vel dignissim magna turpis id sapien. Duis commodo sem id quam. Phasellus dolor. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos.

Chapter 3

Computer Vision applied to Traffic Analysis - An Overview

Este capítulo deve começar por fazer uma apresentação detalhada do problema a resolver[1] podendo mesmo, caso se justifique, constituir-se um capítulo com essa finalidade.

Deve depois dedicar-se à apresentação da solução sem detalhes de implementação. Dependendo do trabalho, pode ser uma descrição mais teórica, mais "arquitetural", etc.

3.1 Secção Exemplo

Neste capítulo apresentam-se exemplos de formatação de figuras e tabelas, equações e referências cruzadas.

Apresenta-se de seguida um exemplo de equação, completamente fora do contexto:

$$CIF_1 : \quad F_0^j(a) = \frac{1}{2\pi\imath} \oint_\gamma \frac{F_0^j(z)}{z-a} dz \tag{3.1}$$

$$CIF_2 : \quad F_1^j(a) = \frac{1}{2\pi\imath} \oint_\gamma \frac{F_0^j(x)}{x-a} dx \tag{3.2}$$

Na Equação 3.2 lorem ipsum dolor sit amet, consectetuer adipiscing elit. Suspendisse tincidunt viverra elit. Donec tempus vulputate mauris. Donec arcu. Vestibulum condimentum porta justo. Curabitur ornare tincidunt lacus. Curabitur ac massa vel ante tincidunt placerat. Cras vehicula semper elit. Curabitur gravida, est a elementum suscipit, est eros ullamcorper quam, sed cursus velit velit tempor neque. Duis tempor condimentum ante.

Phasellus imperdiet, orci vel pretium sollicitudin, magna nunc ullamcorper augue, non venenatis dui nunc quis massa. Pellentesque dolor elit, dapibus venenatis, viverra ultricies, accumsan cursus, orci. Aliquam erat volutpat. Mauris ornare tristique leo. Maecenas eros. Curabitur velit

[1] Na introdução a apresentação do problema foi breve.

nunc, tincidunt vitae, dictum posuere, pulvinar nec, diam. In suscipit mauris a nunc. Pellentesque gravida. Morbi quam lacus, pretium eget, tincidunt vulputate, interdum sed, turpis. Curabitur quis est. Sed lectus lorem, congue vel, dignissim laoreet, blandit a, nisi. Aenean nunc ligula, tincidunt eu, hendrerit vel, suscipit non, erat. Aliquam gravida. Integer non pede. In laoreet augue id leo. Mauris placerat.

A arquitetura do visualizador assenta sobre os seguintes conceitos basecitekn:ZPMD97:

- **Componentes** — Suspendisse auctor mattis augue *push*;
- **Praesent** — Sit amet sem maecenas eleifend facilisis leo;
- **Pellentesque** — Habitant morbi tristique senectus et netus.

3.1.1 Exemplo de Figura

É apresentado na Figura 3.1 um exemplo de figura flutuante.

[t]

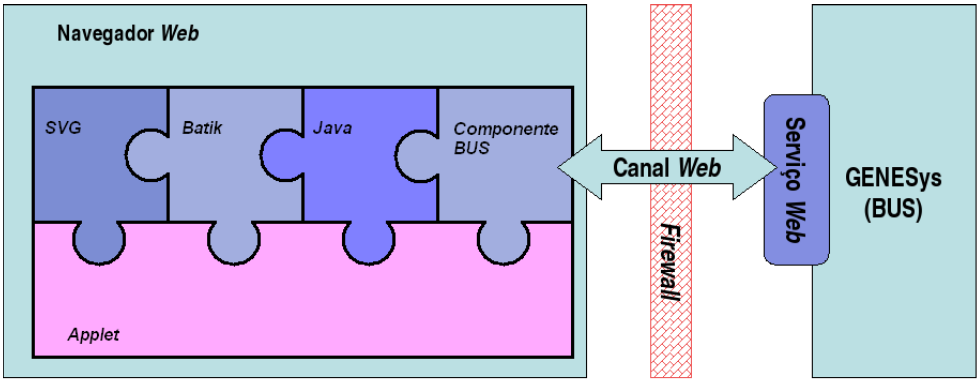


Figure 3.1: Arquitectura da Solução Proposta

Loren ipsum dolor sit amet, consectetuer adipiscing elit. Praesent sit amet sem. Maecenas eleifend facilisis leo. Vestibulum et mi. Aliquam posuere, ante non tristique consectetuer, dui elit scelerisque augue, eu vehicula nibh nisi ac est. Suspendisse elementum sodales felis. Nullam laoreet fermentum urna.

Duis eget diam. In est justo, tristique in, lacinia vel, feugiat eget, quam. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Fusce feugiat, elit ac placerat fermentum, augue nisl ultricies eros, id fringilla enim sapien eu felis. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Sed dolor mi, porttitor quis, condimentum sed luctus.

3.1.2 Exemplo de Tabela

É apresentado na Tabela 3.1 um exemplo de tabela flutuante e na Tabela 3.2 um exemplo de tabela flutuante, um pouco mais complicada.

Table 3.1: Uma Tabela Simples

Acrónimo	**Significado**
ADT	*Abstract Data Type*
ANDF	*Architecture-Neutral Distribution Format*
API	*Application Programming Interface*

Integer quis pede. Fusce nibh. Fusce nec erat vel mi condimentum convallis. Sed at tortor non mauris pretium aliquet. In in lacus in dolor molestie dapibus. Suspendisse potenti. Pellentesque sagittis porta erat. Mauris sodales sapien id augue. Nam eu dolor. Donec sit amet turpis non orci rhoncus commodo. Etiam condimentum commodo libero.

Mauris pede. Curabitur faucibus dictum nibh. Proin tincidunt diam vitae mauris. Sed hendrerit dolor vel ipsum. Nullam dapibus. Vivamus tellus diam, egestas sit amet, vulputate non, vulputate id, eros. Nunc sit amet nibh eget nibh imperdiet ornare. Cras vehicula mattis ipsum. Sed diam arcu, semper at, gravida vitae, fermentum et, nulla. Aenean massa orci, tristique nec, rutrum id, fringilla eget, erat. Curabitur nulla ipsum, aliquam sed, rutrum vitae, semper quis, ante. Fusce at nunc in dolor condimentum tempor. Duis sit amet massa.

Curabitur convallis nulla quis risus. Nulla mollis porttitor purus. Fusce ultricies odio at ligula pellentesque suscipit. Nulla velit libero, blandit a, aliquet quis, hendrerit id, arcu. Phasellus porttitor porttitor purus. Suspendisse velit tortor, fringilla sit amet, commodo a, ultrices et, mi. Donec eu metus in erat ornare adipiscing. Praesent varius mi ac nunc. Vestibulum leo lacus, elementum in, vestibulum sit amet, hendrerit at, justo. Sed sit amet neque. Donec libero risus, commodo sit amet, dignissim ut, tincidunt a, eros. Ut non lacus quis tortor mattis ullamcorper. Vivamus consequat augue vel erat. Sed tincidunt. Sed leo eros, ornare a, pulvinar non, mattis quis, nibh. Aliquam faucibus mi ac nisi.

Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Duis aliquet, libero sit amet ornare viverra, augue erat interdum dolor, vitae tincidunt lorem erat a lacus. Sed lectus nisi, auctor in, hendrerit a, molestie vel, lectus. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Duis lacinia tempor dui. Vivamus rhoncus, tellus a viverra dignissim, pede dui adipiscing odio, non faucibus metus mi gravida eros. Nullam a tellus ut velit elementum tempus. Aenean rutrum convallis tellus. Vestibulum nulla ante, dapibus ut, lobortis ut, varius sed, nisl. Fusce lobortis. Sed ac lorem. Nulla tincidunt nulla eget leo. Maecenas ac lectus eu neque ultrices pharetra. Curabitur a risus nec arcu placerat tempor. Suspendisse magna nisl, viverra a, adipiscing eget, ornare ultricies, ligula. Maecenas eu ligula vitae eros convallis dignissim.

Loren ipsum dolor sit amet, consectetuer adipiscing elit. Praesent sit amet sem. Maecenas eleifend facilisis leo. Vestibulum et mi. Aliquam posuere, ante non tristique consectetuer, dui elit scelerisque augue, eu vehicula nibh nisi ac est. Suspendisse elementum sodales felis. Nullam laoreet fermentum urna.

Table 3.2: Uma Tabela Mais Complicada

Iteração k de $f(x_n)$				
$\boldsymbol{k}$	x_1^k	x_2^k	x_3^k	comentários
0	-0.3	0.6	0.7	-
1	0.47102965	0.04883157	-0.53345964	$\delta < \varepsilon$
2	0.49988691	0.00228830	-0.52246185	$\delta < \varepsilon$
3	0.49999976	0.00005380	-0.523656	N
4	0.5	0.00000307	-0.52359743	
$\vdots$	$\vdots$	$\ddots$	$\vdots$	
7	0.5	0.0	**-0.52359878**	$\delta < 10^{-8}$

Duis eget diam. In est justo, tristique in, lacinia vel, feugiat eget, quam. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Fusce feugiat, elit ac placerat fermentum, augue nisl ultricies eros, id fringilla enim sapien eu felis. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Sed dolor mi, porttitor quis, condimentum sed luctus.

3.2 Secção Exemplo

Loren ipsum dolor sit amet, consectetuer adipiscing elit. Praesent sit amet sem. Maecenas eleifend facilisis leo. Vestibulum et mi. Aliquam posuere, ante non tristique consectetuer, dui elit scelerisque augue, eu vehicula nibh nisi ac est. Suspendisse elementum sodales felis. Nullam laoreet fermentum urna.

Duis eget diam. In est justo, tristique in, lacinia vel, feugiat eget, quam. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Fusce feugiat, elit ac placerat fermentum, augue nisl ultricies eros, id fringilla enim sapien eu felis. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Sed dolor mi, porttitor quis, condimentum sed luctus.

3.3 Resumo e Conclusões

Resumir e apresentar as conclusões que se podem tirar no fim deste capítulo.

Chapter 4

Implementation

Este capítulo pode ser dedicado à apresentação de detalhes de nível mais baixo relacionados com o enquadramento e implementação das soluções preconizadas no capítulo anterior. Note-se no entanto que detalhes desnecessários à compreensão do trabalho devem ser remetidos para anexos. Dependendo do volume, a avaliação do trabalho pode ser incluída neste capítulo ou pode constituir um capítulo separado.

Capitulo implementação Como resolver os desafios Estabilizaçao BG Sub

As previously stated, the work done for this project consists of a module for the "Video Server" being developed by LIACC, that will perform all the required analytics of the video received. This chapter provides the details of the implementation of this module, developed during the dissertation semester between February and June 2017. Tasks performed involved designing an appropriate architecture, able to scale with the project, treating the images received from the video streams to detect events and extract information.

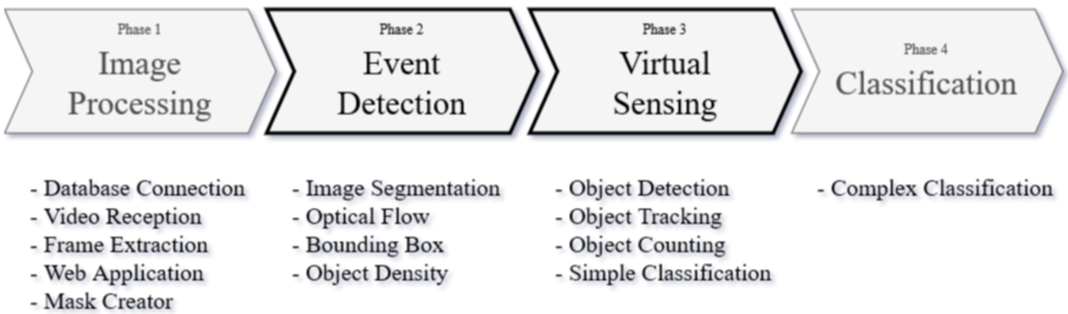


Figure 4.1: Project Planning

It is expected that this chapter provides some insight on how the theoretical backgrounds presented before were used during the project development, namely phase 2 and 3 of the presented project planning in 4.1. Phase 2 focused on the detection of relevant events to the traffic controller such as suspicious approximation to the camera, intrusion in prohibited areas, fallen objects on the road and wrong way travelling, while phase 3 aimed to count and classify vehicles in both urban and high-way scenarios.

4.1 Technology

In order to build this project we needed both a library of already implemented computer vision algorithms as well as a simple way to retrieve frames from both video streams and files. This section describes what were the chosen technologies including a brief description and why it was chosen.

4.1.1 OpenCV

OpenCV is a library composed of implementations of useful computer vision algorithms implementation, widely used across the industry and academy. It has interfaces for multiple programming languages, like C++, Java and Python, but is natively written in C++ in order to take advantage of low level performance enhancements, as performance is an important factor in real time computer vision applications [Ope17a].

The library contains over 2500 algorithms ranging from the more basic image processing such as filtering, morphology operators and geometric transformations to more complex ones that are able to, among others, compare images, track features and camera movements and recognize faces. Along with the image processing capabilities OpenCV also ships with interfaces to stereo cameras such as Kinect that allow users to retrieve a cloud of 3D points and a depth map from the captured image. This was the chosen library as there existed already previous work at LIACC using it, which could be leveraged for this project.

4.1.2 JavaCV

Adicionar referencias

JavaCV is a wrapper for OpenCV written in Java that works on top of the JavaCPP Presets, a project that provides Java interfaces for commonly used C++ libraries, such as OpenCV and FFmpeg, the ones we are using, as well as CUDA, ARToolKitPlus and others. It provides access to all the functionalities of OpenCV inside a Java environment, and was the chosen solution as there was already experience inside LIACC working with this technology.

Even though the code is written in Java and runs inside a Java Virtual Machine, the code from OpenCV is compiled from C/C++ and the memory of the objects created there is allocated in a separate thread. This made it impossible to rely on the garbage collector to do the memory management, and necessary to manually delete the native objects. Failure to address this issue causes the system to run out of memory and a subsequent program crash.

4.1.3 FFmpeg

FFmpeg is a framework that performs encoding, decoding, transcoding, streaming and filter operations on videos in all the well known video formats, across a large number of platforms. It is used

```
// videopath can be either a stream path or a file path
FFmpegFrameGrabber _frameGrabber = FFmpegFrameGrabber.createDefault(videopath);
Frame currentFrame = null;

while( (currentFrame = _frameGrabber.grabImage()) != null) {
  // Use grabbed frame
}
```

Listing 4.1: FFmpegFrameGrabber usage example

in the "Video Server" to read streams from any format and restream them all to the same format. In the Analytic module it is used to grab the frames from both the video files and the streams using the same code, as shown in 4.1.

4.2 Architecture

4.2.1 Implementation Architecture

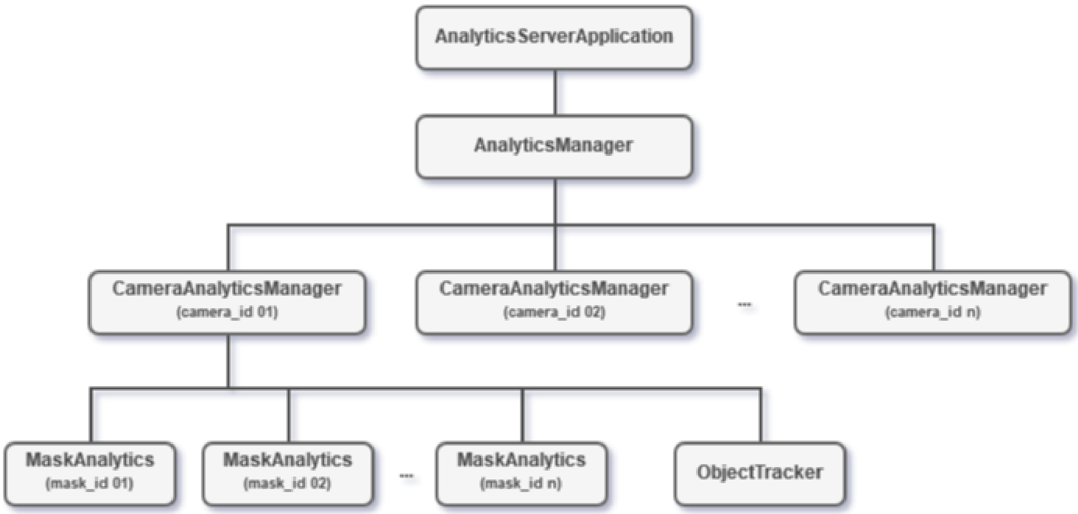


Figure 4.2: Manager Architecture

The analytics module was designed to comply with the following specifications, imposed by both our partner and to comply with the already developed modules.

- Run uninterruptedly waiting for requests to be made
- Process a large number of video inputs at the same time
- Receive input from streams or files
- Specify which analysis are to be run on a specific video

Mais requirements?

The application main thread runs an instance of the AnalyticsManager class that implements the Java Runnable interface. This thread will launch one CameraAnalyticsManager instance for each video to be analysed, be it from stream or from file, and keep track of each worker status, disposing of them when they finish, and launching new ones when a request is received. The CameraAnalyticsManager is then responsible to query the database in order to find out what are the analytics the user wants to retrieve from the video, information that is stored in the *camera* table of the database. This process then starts a frame grabber that will convert each frame of the video into its representation in OpenCV and feed it to each Analyser through a queue. This enables each one of these workers to run at his own pace and if one of them runs slower than the pace at which the frame grabber reads the images, it will simply queue them up and not slow down the remaining workers. The drawbacks of this solution are that it is theoretically possible to run out of memory to keep these frames, although this limit was not reached during the testing phase.

4.3 Segmentation

This section describes how the segmentation of the received images is performed, and how it returns a foreground mask representing the moving areas of the scene.

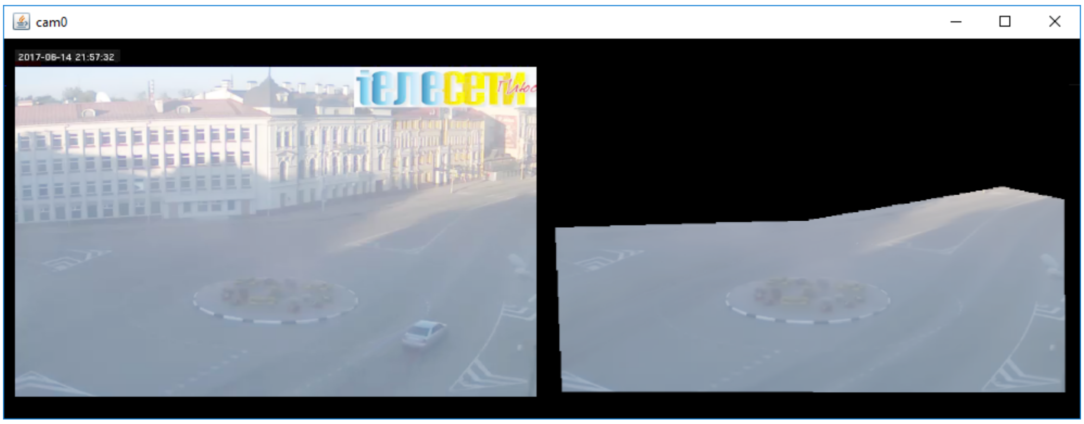


Figure 4.3: Background Subtraction - Background Model

Explain masking process

In figure 4.3 we can see the original frame on the left and the calculated background model of the scene on the right. To achieve this result, the first step is to mask the obtained frame in order to prevent uninteresting regions of the image from being processed the Background Subtracter. This solves an issue where moving or changing regions of the image outside our region of interest would create unwanted artefacts, for example moving trees due to the wind blowing, or the issue that occurred in this scene, where a car passing would be reflected in the windows of the building.

The next step of the Segmentation process is to feed the masked frames into a Background Subtraction algorithm that will use them to update its internal representation of the scene. This project uses the OpenCV implementation of the Mixture of Gaussians algorithm that allows the user to tune:

- Number of past frames considered on the background calculation
- Threshold value from which a pixel is considered to be foreground, compared to the difference between its current value and the one from the background model
- Learning rate of the algorithm, how much each new frame influences the model

After updating the background model we can retrieve from the Background Subtracter its foreground mask, a rough representation that is calculated by subtracting the background model from the current image and thresholding it, thus returning only the pixels where the difference is significant enough.

Figure 4.4: Background Subtraction - Foreground Mask

As we can see in the middle image of figure 4.4 the foreground mask from the Background Subtracter can have a lot of noise due to lighting conditions. To solve this issue two morphology filters are applied to the image, a squared erosion filter to remove the small speckles that appear in the mask, followed by a larger circular dilation filter to consolidate the positive regions of the mask, as the wind shield and windows of the vehicles are usually detected as background due to their dark colour and/or reflection of the environment.

4.4 Object Detection

This section describes how the information about the objects' size and position are retrieved from the binary mask returned from the segmentation process.

OpenCV provides BlobDetector [Ope17b], a family of functions that retrieve information of blobs in an image. It works by thresholding the image into multiple binary images using a range of threshold values, grouping the connected white pixels at each one of these images in binary blobs and then merging blobs that are close enough to each other into larger blobs. This method allows users to filter returned blobs based on 3 properties:

- Area - The area of the region

- Circularity - Ratio between the area of the smallest involving circle and the area of the region
- Convexity - Ratio between the are of the convex hull of the region and its area
- Inertia - Elongation of the region (0 for lines, 1 for circles)

This however does not work as intended for our input, an already binarized image where we just need to group the connected pixels. For this the project uses the OpenCV function *connectedComponentsWithStats*, that retrieves the groups of connected pixels in a binary image along with their area, width, height and both the leftmost and topmost coordinate of the group's bounding box. From this data we can create an ImageObject instance that will represent a moving object in the scene, using the regions with a region size above a given threshold, in order to prevent detection of small objects or noise that got through the filtering process.

Italico para nomes de funçoes?

4.5 Object Tracking

In order for the solution to correctly analyse the objects of the scene it needs to track them during their lifetime, in other words, establish a relation between the objects identified on a frame with the objects identified in the next frame. This is done iterating through each one of the new objects and finding which existing object is closer to him. If the distance between them is smaller than the size of the existing object then they are matched.

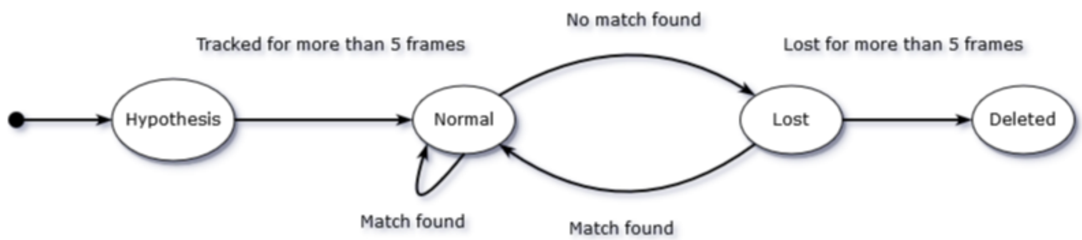


Figure 4.5: Object Life Cycle State Machine

To enhance this mechanism a sub-set of the state machine presented which allows us to model the life cycle of an object. As seen in 4.5 an object starts as an Hypothesis, and cannot be used for analytics while in this state. After being tracked for 5 frames it then changes its state to Normal, where it stays until no match can be found, either from leaving the scene or due to problems with the input frame. An object in this state remains for a maximum of 5 frames, after which it is deleted or until a new object is found in the vicinities of it's last known position.

Referenciar artigo da state machine

This technique relies on objects appearing close to their previous position in the next scene, and because the project was designed to run in high-way scenarios, where vehicles travel at high speeds, there was the need to improve it. The chosen solution was to follow a method used by Chris Dahms in his vehicle counting software [Dah17] that predicts an object next position by calculating an weighted average of the previous inter-position deltas and summing it to its current position as shown in 4.1.

$$NextPos = CurrentPos + \sum_{i=1}^{5} (PosHist(i) - PosHist(i-1)) * (5-i) \tag{4.1}$$

In image 4.6 we can see the result of the tracking process on the right most image. Each detected object is circled in a colour representing its current state in the state machine, blue for normal and red for lost, and shows the trail represents its path across the scene.

Figure 4.6: Object Tracking

4.6 Object Classification

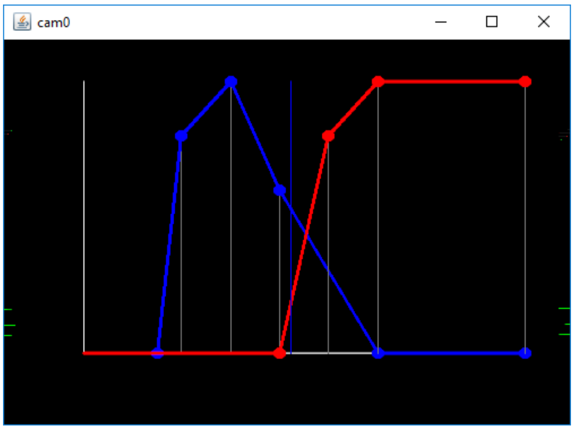


Figure 4.7: Object Classifier

In order to classify the objects present in the scene into light or heavy vehicles a fuzzy set is used. This set is calculated at run time based on a mask drawn by the user via the web interface that roughly approximates the size of a light vehicle. The area of that mask (A_{Light}) is used as a base value to create the fuzzy set shown in figure 4.7. This set has 2 series, one for light vehicles, drawn in blue, and one for heavy vehicles, drawn in red.

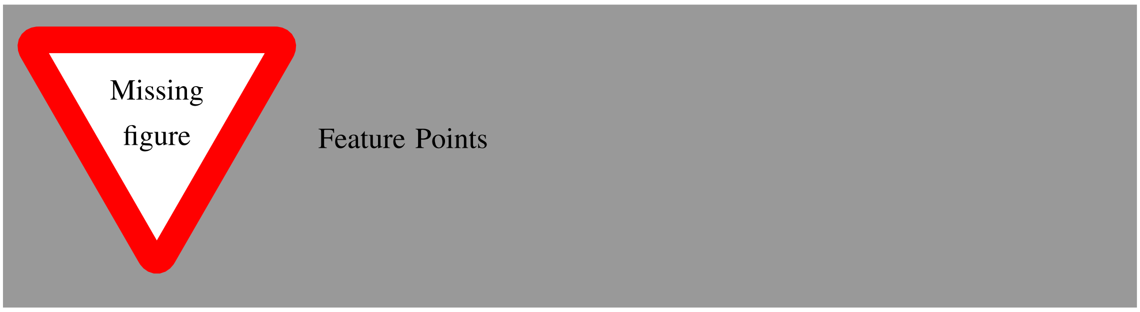
The blue series peaks at A_{Light}, where we have 100% certainty that a matched object is a light vehicle. The point immediate points are at $2/3*A_{Light}$, where the trust is 80% and at $4/3*A_{Light}$

where it drops to 60%. Any object with area below 1/2*A_{Light} or above 2*A_{Light} are considered to have 0% chance to be light vehicles.

The red series peaks at 2*A_{Light}, and any object whose area is larger than this value is considered to be an heavy vehicle with 100% confidence. At the same time, any object whose area is smaller than 4/3*A_{Light} is never considered to be an heavy vehicle, from where the certainty rises to 80% at 5/3*A_{Light}.

When an object is counted in one of the virtual sensors its area is passed to this classifier where an interpolation is calculated for each series, returning the certainty with which it belongs to each one of the classes. Using these values the process can distinguish classifications with certainty below and above a user specified threshold to be treated separately.

4.7 Feature Point Tracking



The need to track feature points stems from the fact that without them we rely wholly on the background subtraction to process the image. Using feature points we can extract and track details of the vehicles that would otherwise be lost due to the process only working with a binary mask.

We can use these points to distinguish vehicles grouped in a single binary region, as they are characterized by This way, even if a vehicle stops for a long time and starts to be considered background we can use its tracked points to maintain the position of

Finish

4.8 Utility

Over the development of the project the rise needed to implement some functionalities to enhance the code flow and to help the visualization of the implemented algorithms. This section will describe the most relevant ones for the reader to analyse and implement in its own projects.

4.8.1 Video Wall

4.8.2 Image Processors

4.9 Summary

Although dealing with a framework with no proper documentation was a difficult challenge to overcome, the performance obtained from the native code was essential for the final result, as the software is now able to track objects in a busy scene faster than the camera's capture rate, allowing us to process a video file in a shorter period of time than it's length. This was a request

Chapter 5

Conclusões e Trabalho Futuro

Deve ser apresentado um resumo do trabalho realizado e apreciada a satisfação dos objetivos do trabalho, uma lista de contribuições principais do trabalho e as direções para trabalho futuro. A escrita deste capítulo deve ser orientada para a total compreensão do trabalho, tendo em atenção que, depois de ler o Resumo e a Introdução, a maioria dos leitores passará à leitura deste capítulo de conclusões e recomendações para trabalho futuro.

Capitulo conclusão Continuaçao para tornar cada parte mais robustas (cada um pode ser uma tese)

This chapter presents an overview of the work done, as well as an evaluation of the state of completion achieved during the semester, followed by a list of the contributions and a foresight into the next steps of the project.

5.1 Satisfação dos Objetivos

5.2 Trabalho Futuro

Referir agrupamento com base no match n para n

Contagem tempo em video

Analise cruzamentos

References

[ACM12] ACM. The 2012 ACM Computing Classification System — Association for Computing Machinery, 2012.

[Aya98] N. Ayache. Medical image analysis a challenge for computer vision research. In *Proceedings. Fourteenth International Conference on Pattern Recognition (Cat. No.98EX170)*, volume 2, pages 1255–1256 vol.2, August 1998.

[BP98] Jorge Badenas and Filiberto Pla. Applying computer vision techniques to traffic monitoring tasks. In *Methodology and Tools in Knowledge-Based Systems*, pages 776–785. Springer, Berlin, Heidelberg, June 1998.

[BT73] T. O. Binford and J. M. Tenenbaum. Computer vision. *Computer*, 6(5):19–24, May 1973.

[BVO11] N. Buch, S. A. Velastin, and J. Orwell. A Review of Computer Vision Techniques for the Analysis of Urban Traffic. *IEEE Transactions on Intelligent Transportation Systems*, 12(3):920–939, September 2011.

[CGPP03] R. Cucchiara, C. Grana, M. Piccardi, and A. Prati. Detecting moving objects, ghosts, and shadows in video streams. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(10):1337–1342, October 2003.

[Dah17] Chris Dahms. OpenCV_3_Car_Counting_Cpp, May 2017.

[DW84] K. W. Dickinson and R. C. Waterfall. IMAGE PROCESSING APPLIED TO TRAFFIC, 1-A GENERAL REVIEW. *Traffic Engineering & Control*, 25(1), January 1984.

[HBJ+16] K. Hammoudi, H. Benhabiles, A. Jandial, F. Dornaika, and J. Mouzna. Self-driven and direct spatio-temporal mechanisms for the vision-based parking slot surveillance. In *2016 SAI Computing Conference (SAI)*, pages 1327–1329, July 2016.

[HK12] M. F. Hashmi and A. G. Keskar. Analysis and monitoring of a high density traffic flow at T-intersection using statistical computer vision based approach. In *2012 12th International Conference on Intelligent Systems Design and Applications (ISDA)*, pages 52–57, November 2012.

[Hua96] T. Huang. Computer Vision : Evolution And Promise, 1996.

[LB78] Michael L. Baird. SIGHT-I: A Computer Vision System for Automated IC Chip Manufacture. *IEEE Transactions on Systems, Man, and Cybernetics*, 8(2):133–139, February 1978.

[LBT17] Robert P. Loce, Raja Bala, and Mohan Trivedi. Detection of Passenger Compartment Violations. In *Computer Vision and Imaging in Intelligent Transportation Systems*, pages 432–. Wiley-IEEE Press, 2017.

[LCC12] X. Li, Q. Chen, and H. Chen. Detection and tracking of moving object based on PTZ camera. In *2012 IEEE Fifth International Conference on Advanced Computational Intelligence (ICACI)*, pages 493–497, October 2012.

[LKR+16] G. Lira, Z. Kokkinogenis, R. J. F. Rossetti, D. C. Moura, and T. Rúbio. A computer-vision approach to traffic analysis over intersections. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pages 47–53, November 2016.

[LV01] B. P. L. Lo and S. A. Velastin. Automatic congestion detection system for underground platforms. In *Proceedings of 2001 International Symposium on Intelligent Multimedia, Video and Speech Processing. ISIMP 2001 (IEEE Cat. No.01EX489)*, pages 158–161, 2001.

[Ope17a] OpenCV. About - OpenCV library, 2017.

[Ope17b] OpenCV. OpenCV: Cv::SimpleBlobDetector Class Reference, 2017.

[Pic04] M. Piccardi. Background subtraction techniques: A review. In *2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No.04CH37583)*, volume 4, pages 3099–3104 vol.4, October 2004.

[SJK09] Y. Sheikh, O. Javed, and T. Kanade. Background Subtraction for Freely Moving Cameras. In *2009 IEEE 12th International Conference on Computer Vision*, pages 1219–1225, September 2009.

[SLZ16] C. Supriyanto, A. Luthfiarta, and J. Zeniarja. An unsupervised approach for traffic sign recognition based on bag-of-visual-words. In *2016 8th International Conference on Information Technology and Electrical Engineering (ICITEE)*, pages 1–4, October 2016.

[SMG+93] J. W. Snell, M. B. Merickel, J. C. Goble, J. B. Brookeman, and N. F. Kassell. Model-based segmentation of the brain from 3-D MRI using active surfaces. In *1993 IEEE Annual Northeast Bioengineering Conference*, pages 164–165, March 1993.

[SS01] Linda G. Shapiro and George C. Stockman. *Computer Vision*. Prentice Hall, Upper Saddle River, NJ, 2001.

[SS11] D. Soendoro and I. Supriana. Traffic sign recognition with Color-based Method, shape-arc estimation and SVM. In *Proceedings of the 2011 International Conference on Electrical Engineering and Informatics*, pages 1–6, July 2011.

[Sto80] K. J. Stout. Computer vision and robots. *Production Engineer*, 59(4):9–, April 1980.

[Ziv04] Z. Zivkovic. Improved adaptive Gaussian mixture model for background subtraction. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, volume 2, pages 28–31 Vol.2, August 2004.

[ZY14] Daniya Zamalieva and Alper Yilmaz. Background subtraction for the moving camera: A geometric approach. *Computer Vision and Image Understanding*, 127:73–85, October 2014.