Proposta de aplicativo para controle de fluxo de trânsito usando Arduino e câmera com OPENCV

Márcio Takano[1]; Luiz Fernando Braga Lopes[2]

Departamento de Pós-graduação - Faculdade Cidade Verde (FCV)

Maringá – PR

{takanomarcio@gmail.com, prof_braga@fcv.edu.br}

Abstract: *This article presents a proposal for an application to control the traffic flow in a city using the Arduino platform with the camcoder integration using resources of the OpenCV library, to assist in traffic congestion prevention and disorders generated in the flow of vehicles especially in peak hours or on certain routes of a city, helping to better security and management in the control flow Keywords: Applications; Flow; OpenCV; Traffic; Vehicles*

Resumo: *Este artigo apresenta uma proposta de um aplicativo para controlar o fluxo de trânsito em uma cidade usando a plataforma Arduino com a integração de câmeras utilizando recursos da biblioteca OpenCV, para auxiliar na prevenção de congestionamentos e transtornos gerados no fluxo de veículos principalmente nos horários de picos ou em determinadas rotas de uma cidade, ajudando assim a uma melhor segurança e gerenciamento no controle de fluxo. Palavras-Chave: Aplicativos; Fluxo; OpenCV; Trânsito; Veículos.*

1. Introdução

O trânsito das grandes metrópoles, nos dias atuais, passa por uma situação caótica, presenciamos e observamos sérios problemas com fluxo de veículos, congestionamentos intensos, demoras no tráfego de veículos, problemas que muitas vezes, são causados por falta de manutenção adequada ou um planejamento fora da realidade vivida no transito, também ocorre a imprevisão ou fatos isolados como, a interferência da natureza e colisões em vias principais. Outro ponto crítico vivido no trânsito das grandes metrópoles é a falta de segurança nas vias públicas, através de sequestros relâmpagos, perseguições, ou roubos em sinais de trânsito. Desta forma surge então a necessidade de meios que possibilitem a melhoria de gerenciamento e controle das vias principais, ou em vias que contêm um grande fluxo de veículos. Para tal necessidade este artigo sugere uma proposta tecnológica utilizando câmeras de monitoramento online com o uso do recurso OpenCV. Desta forma auxiliando no controle e gerenciamento do fluxo de trânsito, na qual serão ajustados por um parâmetro o tempo para controlar a abertura e fechamento dos semáforos, através deste recurso também haverá a possibilidade de detectar a placa, cor e tamanho de veículo em circulação. Com isso autoridades de trânsito conseguirão gerenciar infrações e delitos ocorridos nas vias. Neste modelo será utilizado a plataforma Arduino que possui diversas funções e recursos para auxiliar na integração com o monitoramento online.

2. Revisão Bibliográfica

A seguir será apresentado alguns problemas detectados no trânsito das grandes metrópoles, será comentado também sobre os recursos tecnológicos empregados para a construção do protótipo proposto no artigo.

3. Trânsito

Com o surgimento dos veículos motorizados a partir do início do século XX o trânsito nas grandes metrópoles, sofreu com um crescimento de forma muito acelerada e ainda sofre, até os dias atuais.

Segundo o jornal [Gazeta do povo] o trânsito de Maringá, cidade que se encontra na região norte do Estado do Paraná está à beira de um colapso, apesar da cidade possuir largas avenidas que comportam um número grande de veículos em circulação. Pessoas que circulam pelas avenidas durante os horários de picos tem que ter paciência para conseguir avançar poucos metros em um considerável período de tempo. Segundo [Maria M. P. 2015] nos últimos anos a frota de veículos na cidade alcançou a média de 1,6 veículos por cada habitante, um número considerado alto para uma população de quase 400 mil habitantes.

Uma série de situações de conflitos e tumultos nas grandes cidades brasileiras e mundiais vem ocorrendo. Podemos citar alguns casos de congestionamentos exagerados, acidentes ocorridos com elevada frequência, transporte público deficiente, ruas planejadas de forma irregular, motorista saturados e impacientes, trabalhadores que atrasam em seus compromissos nas empresas por questão de ficar preso no transito, violência, por questão de stress e nervosismo, entre outros. Fatores que se agravam com o decorrer dos anos. Órgãos governamentais não tomam a iniciativa preventiva para a solução desses e de outros problemas, "muito se promete e pouco se faz". Mas quais soluções são necessárias para solução desses problemas. Em São Paulo já contamos com o sistema de rodízios, mas ainda ocorre problemas no transporte público, existem também imóveis em áreas centrais desocupados, levando a população procurarem moradias nas áreas periféricas. Já em Maringá temos a deficiência no controle da sincronia dos sinais, e também existem vários pontos de estrangulamento das vias que prejudicam o fluxo de veículos de forma correta.

4. Arduino

Segundo [Evans M., e Noble. J. e Hochenbaum J. 2013] o Arduino é uma plataforma criada na Itália na cidade de Ivrea no ano de 2005, seu intuito era baratear e tornar mais fácil para os estudantes de design trabalhar com tecnologia. Seu sucesso foi sinalizado com o recebimento de uma menção honrosa na categoria *Comunidades Digitais* em 2006, pela Prix Ars Electronica. O seu projeto original foi melhorado e novas versões foram introduzidas, sua marca de vendas chega a mais de 3000.000 placas vendidas e elas são vendidas por uma infinidade de fabricantes diferentes.

Temos uma série de versões do Arduino todas baseadas em um micro-controlador Atmel AVR de 8 bits, com componentes complementares para facilitar a programação e incorporação para outros circuitos. Um importante aspecto é a maneira padrão que os conectores são expostos, permitindo o CPU ser interligado a outros módulos expansivos, conhecidos como shields. Os Arduinos originais utilizam a série de chips

megaAVR, especialmente os *ATmega8*, *ATmega168*, *ATmega328* e a *ATmega1280*; porém muitos outros processadores foram utilizados por clones deles.

O benefício que o Arduino proporciona, se dá pela sua facilidade de manuseio e a gama da compatibilidade de dispositivos e módulos que podem ser acoplados na sua prototopagem.

Para trabalhar com o Arduino precisamos de sua IDE (Arduino IDE) que pode ser baixado no site https://www.arduino.cc, sua vantagem é por ser open-source, ou seja sem custo de licença, no entanto existem outras IDEs no mercado. A linguagem de programação para Arduino, se baseia na linguagem C++. Sua IDE já vem com algumas bibliotecas prontas para uso, e alguns exemplos práticos, ajudando no desenvolvimento para usuários que estão iniciando no desenvolvimento desta plataforma. A conexão com o Computador se dá por entrada USB, através de uma porta de comunicação, normalmente usa-se a porta com3.

5. Visão Computacional

Segundo [Souza, C. F.; Capovilla, F.; Eleotério, T. C. 2010] Por visão computacional entente se como sendo a ciência responsável pela visão de uma máquina, sendo como a forma que o computador enxerga o meio a sua volta, extraindo informações importantes de imagens capturadas por câmeras de vídeo, sensores, scanners, Na Visão Computacional temos como um input imagens e através do seu processo tomamos como output modelos matemáticos, ou seja, um agrupamento de técnicas e métodos nos quais se torna possível a interpretação de uma imagem emulando a visão humana. Dessa forma, com a aplicação de tais técnicas é possível então fazer um computador reconhecer suas mãos, reconhecer seus gestos, reconhecer seu rosto, realizar sua identificação por impressão digital, etc.

A aplicação de técnicas de visão computacional em diferentes situações tornou-se possível devido ao aumento da capacidade de processamento dos computadores pessoais. Também a popularização das câmeras digitais permite que usuários de computadores utilizem aplicativos com interfaces baseadas na interação com vídeo. Na figura 1 podemos ver o reconhecimento do objeto em lote, onde por meio de técnicas e algortimos precisos é possível distinguir objetos separadamente sem necessariamente que a imagem possua uma qualidade excepcional.

Figura 1 - Reconhecimento de Objetos em Imagem.

Outra forma de uso de visão computacional se dá pelo uso de reconhecimento facial. Atualmente é comum a utilização deste tipo de tecnologia em câmeras digitais ou smartphones e tablets mais novos. Na figura 2 podemos perceber que é feito o reconhecimento facial dos rostos das pessoas que tem seus rostos detectados com maior clareza, na qual seus rostos são enquadrados, e não existe enquadramento dos rostos que ocorrem dificuldades de reconhecimento.

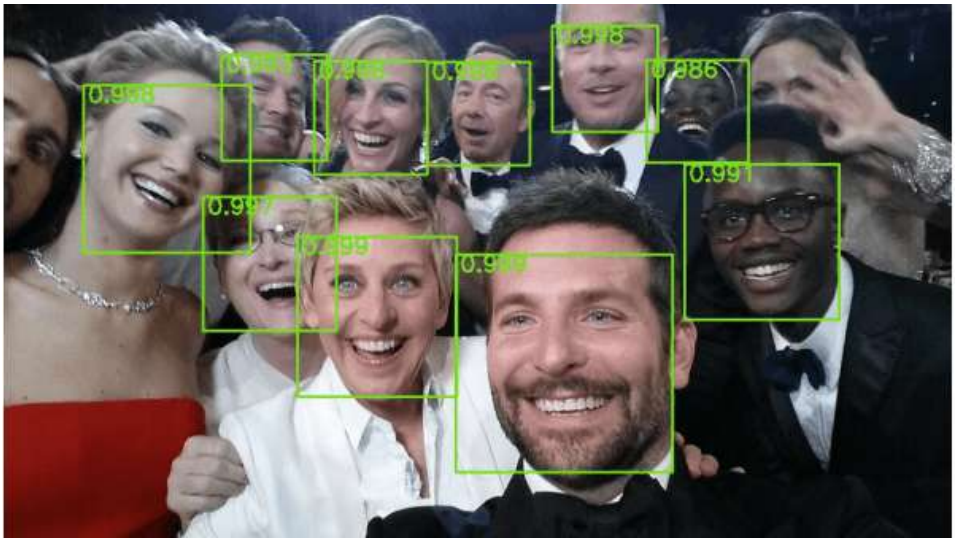

Figura 2 - Detecção de Faces.

Outra forma de utilização da visão computacional está sendo utilizado na área de jogos um exemplo disso é o uso do kinect, usado nos consoles Xbox para controle por gestos e movimentos.

Figura 3 - Tela de interação do Kinect do Xbox.

Pode também desenvolver programas que façam uso de visão computacional. Para isso, uma das principais bibliotecas disponíveis é a OpenCV, originalmente criada pela Intel em 1999, e agora mantida por toda uma comunidade de desenvolvedores, com suporte a linguagens tais como C, C++, Python, Java, dentre outras.

Basicamente, toda imagem é uma **matriz**, cujas linhas e colunas endereçam os **pixels** da referida imagem. Por sua vez, como estes pixels serão referenciados irá depender se a imagem está em escala de cinza, colorida, etc.

3.1. Processo de reconhecimento

A organização de um sistema de visão computacional é dependente da aplicação. A implementação específica de tal sistema depende também se sua funcionalidade é pré-especificada ou se existe alguma parte de aprendizagem durante a operação. Existem, entretanto, funções típicas encontradas vários sistemas de visão computacional:

- **Aquisição de imagem:** uma imagem digital é produzida por um ou vários sensores. Dependendo do tipo do sensor, o resultado pode variar entre uma imagem bidimensional, uma cena tridimensional ou ainda uma sequência de imagens. Os valores dos pixels geralmente indicam a intensidade da luz em uma ou várias faixas de cor (o que forma imagens em tom de cinza ou coloridas), mas também podem indicar valores físicos como profundidade e absorção ou reflexão das ondas eletromagnéticas.
- **Pré-processamento:** antes de um método de visão computacional ser aplicado em uma imagem para extrair informação, é geralmente necessário processar a imagem para assegurar-se que ela satisfaz as condições do método. Exemplos incluem remapeamento (para assegurar o sistema de coordenadas), redução de ruídos (para assegurar que as informações são verdadeiras) e aumento de contraste (para assegurar que as informações relevantes serão detectadas).
- **Extração de características:** características matemáticas da imagem em vários níveis de complexidade são extraídos. Exemplos básicos incluem detecção de bordas, cantos ou pontos. Exemplo sofisticados incluem a morfologia matemática, detecção de texturas, formatos e movimentos.
- **Detecção e segmentação**: em algum ponto do processo uma decisão é feita sobre a relevância de regiões da imagem para processamento posterior. Exemplos incluem a seleção de regiões de interesse específicos e segmentação de uma ou mais regiões que contém um objeto de interesse.
- **Processamento de alto nível:** neste ponto a entrada é geralmente um conjunto pequeno de dados. O processo posterior inclui a verificação da satisfação dos dados, a estimativa de parâmetros sobre a imagem e a classificação dos objetos detectados em diferentes categorias.

6. OpenCV

Nos dias atuais, a utilização de câmeras de monitoramento nas vias públicas, tornou-se realidade, nas grandes metrópoles como, São Paulo e Rio de Janeiro esta tecnologia está sendo empregada na punição em infrações de trânsito, sendo que a autorização deste tipo de equipamento foi sancionada pelo Conselho Nacional de Trânsito (Contran)

em julho de 2015. Com a regulamentação deste tipo de equipamento podemos iniciar estudos com a integração do recurso OpenCV (*Open Source Computer Vision Library*), que é uma tecnologia desenvolvida pela Intel, em 2000, ou seja, uma biblioteca multiplataforma, totalmente livre ao uso acadêmico e comercial, para o desenvolvimento de aplicativos na área de Visão computacional, bastando seguir o modelo de licença BSD Intel. O OpenCV possui módulos de Processamento de Imagens e Video I/O, Estrutura de dados, Álgebra Linear, GUI (Interface Gráfica do Usuário) básica com sistema de janelas independentes, Controle de mouse e teclado, além de mais de 350 algorítmos de Visão computacional como: Filtros de imagem, calibração de câmera, reconhecimento de objetos, análise estrutural e outros. O seu processamento é em tempo real de imagens.

Esta biblioteca foi desenvolvida nas linguagens de programação C/C++. Também, dá suporte a programadores que utilizem Java, Python e Visual Basic e desejam incorporar a biblioteca a seus aplicativos. A versão 1.0 foi lançada no final de 2006 e a 2.0 foi lançada em setembro de 2009.

OpenCV foi uma proposta da Intel Research de melhorar aplicações de uso intensivo de processamento, sendo parte de uma série de projetos que incluíam Ray tracing e monitores 3D. Os principais contribuidores do projeto eram da Intel Russia, assim como o time de desempenho de bibliotecas da Intel. Os objetivos deste projeto foram listados como:

- Avançar a pesquisa de visão, fornecendo código aberto e também código otimizado para visão básica de Infra-estrutura.
- Disseminar o conhecimento da visão, fornecendo uma infra-estrutura comum que os desenvolvedores poderiam construir, onde o código seria mais legíveis e transferíveis. Baseados na visão do avanço de aplicações comerciais, tornando portátil, com desempenho otimizado.
- Código disponível gratuitamente.

6.1. Plataformas compatíveis

OpenCV pode funcionar sobre Windows, Android, Maemo, FreeBSD, OpenBSD, iOS, BlackBerry 10, Linux e OS X.

6.2. Áreas de aplicação

- Humano-Computador Interface (HCI)
- Identificação de objetos
- Sistema de reconhecimento facial
- Reconhecimento de movimentos
- Gravação de vídeos
- Robôs móveis
- Reconstrução 3D
- Realidade Virtual
- Realidade Aumentada
- Realidade Misturada

6.3. Estrutura do OpenCV

- **core** — Módulo de estruturas centrais de dados, tipos de dados e gerenciamento de memória.
- **Imgproc** — Módulo de filtragem de imagens, transformações geométricas imagem, estrutura e análise de forma.
- **Highgui** — Módulo de Controle de Interface, leitura e escrita de imagens e vídeo.
- **Video** — Módulo de Controle de Interface e dispositivos de entrada.
- **Calib3d** — Módulo de calibração da câmara e reconstrução 3D a partir de vários pontos de vista.
- **Features2d** — Módulo de Extração de características, descrição e correspondência.

7. Java

Java é uma linguagem de programação interpretada e orientada à objetos desenvolvida na década de 90, por uma equipe de programadores chefiada por James Gosling, na empresa Sun Microsystems. Diferentemente das linguagens convencionais, que são compiladas para código nativo, a linguagem Java é compilada para um bytecode que é executado por uma máquina virtual. A linguagem de programação Java é a linguagem convencional da Plataforma Java, mas não sua única linguagem.

7.1. Aquisição pela Oracle

Em 2009 a Oracle Corporation adquire a empresa responsável pela linguagem Java, a Sun Microsystems, por US$ 7,4 bilhões. Com o objetivo de levar o Java e outros produtos da Sun a seu portfólio.

7.2. Características

A linguagem Java foi projetada tendo em vista os seguintes objetivos:

- Orientação a objetos - Baseado no modelo de Simular
- Portabilidade - Independência de plataforma - "escreva uma vez, execute em qualquer lugar" ("write once, run anywhere")
- Recursos de Rede - Possui extensa biblioteca de rotinas que facilitam a cooperação com protocolos TCP/IP, como HTTP e FTP
- Segurança - Pode executar programas via rede com restrições de execução

Além disso, podem-se destacar outras vantagens apresentadas pela linguagem:

- Sintaxe similar a C/C++
- Facilidades de Internacionalização - Suporta nativamente caracteres Unicode
- Simplicidade na especificação, tanto da linguagem como do "ambiente" de execução (JVM)
- É distribuída com um vasto conjunto de bibliotecas (ou APIs)
- Possui facilidades para criação de programas distribuídos e multitarefa (múltiplas linhas de execução num mesmo programa)
- Desalocação de memória automática por processo de coletor de lixo
- Carga Dinâmica de Código - Programas em Java são formados por uma coleção de classes armazenadas independentemente e que podem ser carregadas no momento de utilização.

7.3. Bibliotecas de classe do Java

Programas Java consistem em partes chamadas classes. As classes incluem partes chamadas métodos que realizam tarefas e retornam informações quando as tarefas são concluídas. Você pode criar cada parte necessária para formar seus programas Java. Entretanto, a maioria dos programadores Java tira proveito das ricas coleções de classes existentes nas bibliotecas de classe Java, que também são conhecidas como Java APis (Application Programming Interfaces). Portanto, na realidade há dois aspectos para aprender o "mundo" do Java. O primeiro aspecto é a própria linguagem Java, de modo que você possa programar suas próprias classes, o segundo são as classes nas extensas bibliotecas de classe Java. Para baixar a documentação da Java API, acesse java.sun. com/javase/down1oads/, role para baixo até a Seção Additional Resources c clique no botão Download à direita de Java SE 6 Documentation.

8. JavaFX

A plataforma JavaFX é uma plataforma de software desenvolvida em java pela Oracle para a criação e disponibilização de Aplicação Rica para Internet que pode ser executada em vários dispositivos diferentes.

A versão atual (JavaFX 2.1.0), permite a criação para desktop, browser e dispositivos móveis. TVs, video-games, Blu-rays players e outras plataformas estão sendo planejadas para serem adicionadas no futuro. O suporte nos desktops e browsers é através da JRE e nos dispositivos móveis através do JavaME.

Para construir aplicações os desenvolvedores usam uma linguagem estática tipada e declarada chamada JavaFX Script. No desktop existe implementação para Windows(x86/x64), Mac OS X e Linux (X86/X64). Nos dispositivos móveis, JavaFX é capaz de suportar vários sistemas operacionais moveis como Android, Windows Mobile, e outros sistemas proprietários.

A atual versão do JavaFX inclui os seguintes componentes:

1. O JavaFX SDK: Compilador e ferramentas para JavaFX. Gráficos, Media Web e documentos de textos com formatação.
2. NetBeans IDE para JavaFX - Com a ajuda da paleta do Netbeans JavaFX o processo vira somente um "drag-n-drop", efeitos, animações e exemplos. Para eclipse também existe um plugin chamado Kenai [1].
3. As ferramentas e os plugins para programas de criação :Project Nile é um plugin em desenvolvimento para ligar Adobe Photoshop, Adobe Illustrator assim podendo exportar gráficos com o código de JavaFX, ferramentas para converter SVG gráfico em JavaFX Script. Destaques técnicos

Perfil Comum - JavaFX é baseado no conceito 'Common Profile' que representa a reutilização de muita parte do código em todos os dispositivos seja móvel ou desktop. Isto permite aos desenvolvedores usar modelos de programação comum enquanto constrõem para Desktop ou dispositivos Moveis. Para diferenciar as qualidades de cada dispositivos por exemplo o JavaFX 1.1 possui uma API para Desktop que inclui SWING e efeitos visuais avançados.

Integração para criação em programas terceiros - JavaFX inclui plugins para Adobe Photoshop e Adobe Illustrator que permite a criação de gráficos avançados para integrar

diretamente nas aplicações de JavaFX. Os plugins geram códigos em JavaFX Script que preservam o layout e a estrutura dos gráficos. Desenvolvedores podem facilmente adicionar animações e efeitos para os gráficos estáticos importados. Também há um SVG gráfico conversor que permite importar e rever apos ser convertido no formato JavaFX.

9. Eclipse (software)

O Eclipse é um IDE para desenvolvimento Java, mas também suporta outras linguagens a partir de plugins como C/C++,PHP,ColdFusion, Python, Scala e plataforma Android. Ele foi feito em Java e segue o modelo open source de desenvolvimento de software. Atualmente faz parte do kit de desenvolvimento de software recomendado para desenvolvedores Android.

O projeto Eclipse foi iniciado na IBM que desenvolveu a primeira versão do produto e doou-o como software livre para a comunidade. O gasto inicial da IBM no produto foi de mais de 40 milhões de dólares . Hoje, o Eclipse é o IDE Java mais utilizado no mundo. Possui como característica marcante o uso da SWT e não do Swing como biblioteca gráfica, a forte orientação ao desenvolvimento baseado em plug-ins e o amplo suporte ao desenvolvedor com centenas de plug-ins que procuram atender as diferentes necessidades de diferentes programadores.

10. Proposta

Vários estudos foram levantados, como pesquisas do uso de visão computacional na área de robótica, rastreio de objetos, reconhecimento automático de placa de veículos, reconhecimento de pessoas por traços faciais e também o uso de jogos usando visão computacional e com base nas informações pesquisadas torna se mais claro a elaboração da montagem da aplicação proposta, servindo com intuito de minimizar os problemas encontrando atualmente no trânsito nas cidades, para a implementação do modelo será necessário a instalação e configuração da biblioteca OPENCV junto com o JAVAFX. Neste caso utilizaremos a IDE de desenvolvimento JAVA Eclipse.

Na figura abaixo temos um exemplo de captura de imagem usando a biblioteca OPENCV usando JAVAFX.

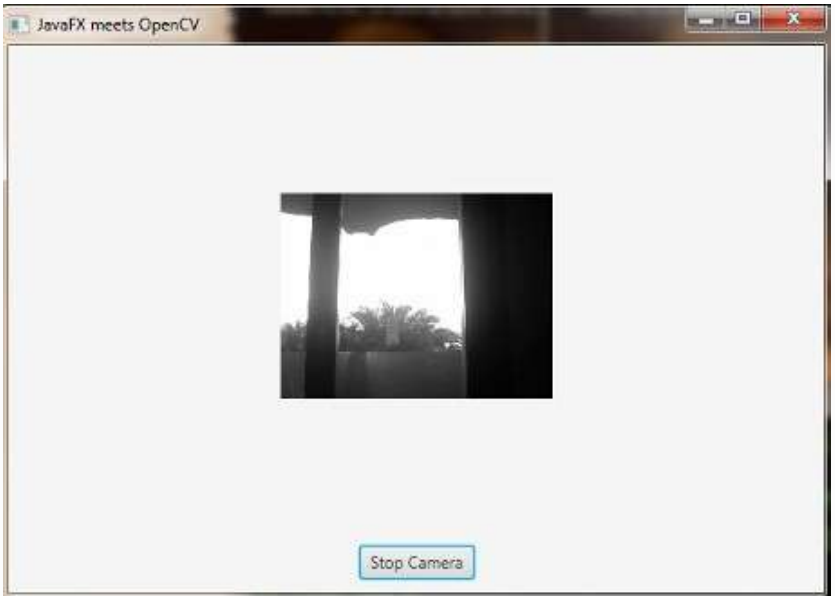

Figura 4. Projeto usando JavaFx com a biblioteca OpenCV.

Com os recursos da utilização da biblioteca OPENCV podemos destacar, sua principal característica a sua versalidade e a enorme gama de recursos que são possibilitados com seu uso. Na Figura 5 podemos exemplificar a utilização de filtragem de dados para a melhoria de qualidade em fotos. Podemos verificar que a imagem original se apresenta de forma escura, dificultando a visualização da placa do veículo, mas com o tratamento correto, pode se notar uma melhoria surpreendente ajustando sua tonalidade e clareamento ideal. Com isso é possível identificar o número de identificação da placa, e até o modelo do veículo.

Figura 5. Diferenças entre imagens no processamento e visão computacional.

Na Figura 6 podemos ter a ideia do verdadeiro poder do recurso de visão computacional. A utilização prática de uma operação de visão computacional é onde o recurso OPENCV consegue fazer a focagem dentro de um quadrante e consegue fazer a distinção e leitura da placa.

Figura 6. Exemplo de leitura de placa usando OPENCV

Na Figura 7 podemos notar um exemplo prático utilizado em rodovias para efetuar a contagem de veículos de um período de tempo. Neste tipo de visão computacional pode se definir pelo tamanho o tipo de objeto em circulação e também definir um ponto de passagem determinada por uma linha reta.

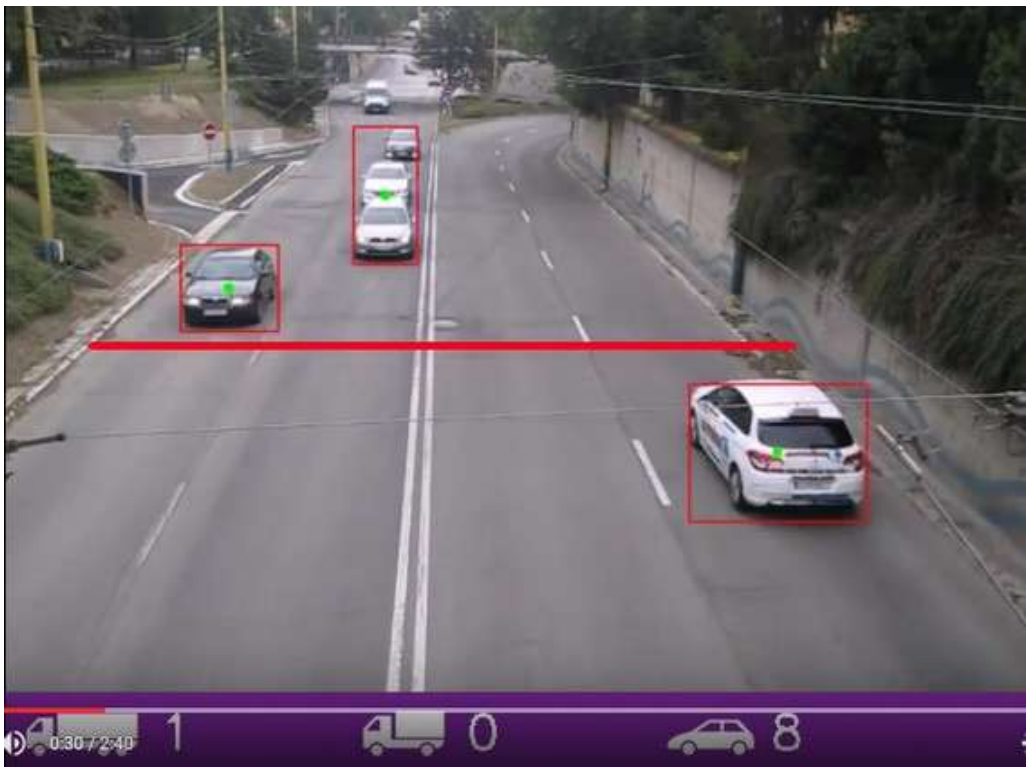

Figura 7. Contagem de veículos por tipo de veículo.

Para a construção do protótipo, foi utilizado Arduino UNO, que por sua vez, também podem ser substituídos por Arduinos NANO, pois estes ocupam menor espaço, precisaremos de led para elaborar os sinais de trânsito, uma protoboard para fazer as ligações dos leds, câmeras e fios para ligações (fios de cabos de rede) e 4 câmeras comuns pode ser de webcam com saída USB e um hub USB para ligações das mesmas. Na Figura 7 segue em detalhes como será a elaboração do protótipo.

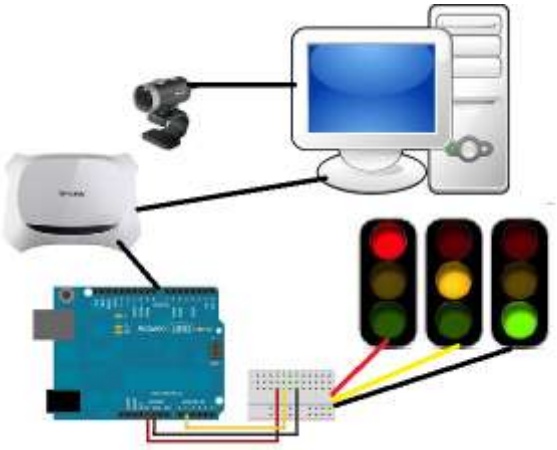

Figura 8. Modelo do Protótipo.

No protótipo, o Arduino está ligado em um roteador através de um módulo ethernet shield que faz a comunicação com um servidor que pegará as informações geradas pelas imagens das câmeras que por sua vez foi feito o tratamento de visão computacional, para contar a quantidade de fluxo de trânsito, onde é passado as informações para o Arduino fazer o controle dos semáforos, controlando a abertura e fechamento dos semáforos conforme o fluxo de trânsito. Este controle é feito usando parâmetros para definir se o fluxo está nos níveis de controle baixo, moderado ou congestionado. Caso estiver congestionado é feito um controle maior de tempo no semáforo para liberar o fluxo, senão o tempo permanece normal.

Na Figura 9 está sendo feito a montagem da maquete, que simulará a forma com que o trânsito é controlado hoje e através de simulações podemos estudar uma melhor forma de customizar e ajustar o tráfego nas horas de maior movimento, ou seja as horas de idas e vindas ao trabalho.

Figura 9. Maquete do Protótipo.

10.1 Construção do Semáforo

Para a simulação do semáforo são necessários alguns acessórios para sua construção:

- Arduíno Uno.
- Protoboard para a montagem dos circuitos elétricos.
- Leds nas cores vermelha, amarela e verde.
- Resistores para proteção de queima dos Leds.
- Fios para conectar os Leds.

Na maquete cada semáforo será inserido em uma via, totalizando 4 semáforos e cada semáforo será constituído por:

3 resistências de 1k.
3 LEDs

- 1 na cor vermelha
- 1 na cor verde
- 1 na cor amarela

Ocupando 3 portas digitais (Vermelho, Amarelo, Verde)

Ferramenta utilizada

- IDE de desenvolvimento para arduíno. https://www.arduino.cc/

Descrição do funcionamento do projeto

O programa inicia com a sequência normal onde cada semáforo é controlado por vez.

Ao acionar o semáforo é enviado um sinal digital de 5 volts que fará com que o led de cor verde acenda, dando a entender que o semáforo está livre para o trafego de

veículos. Após um delay(atraso) de 10 segundos é enviado um sinal digital de 5 volts que fará com que o led de cor verde se apague, e por sua vez o led de cor amarela acenda, dando a entender que o semáforo está em posição de atenção.

Novamente após um delay, desta vez de 5 segundos, é enviado um sinal digital de 5v que fará com que led de cor amarela se apague e acenda o led de cor vermelha, alertando a parar. Assim o próximo semáforo é ativado.

Este mesmo procedimento ocorre nos 4 semáforos quando estão ausentes de alguma funcionalidade específica. Entretanto a aplicação usando OPENCV pode interferir no tempo de abertura e fechamento dos sinais, conforme a situação do fluxo de trânsito da via.

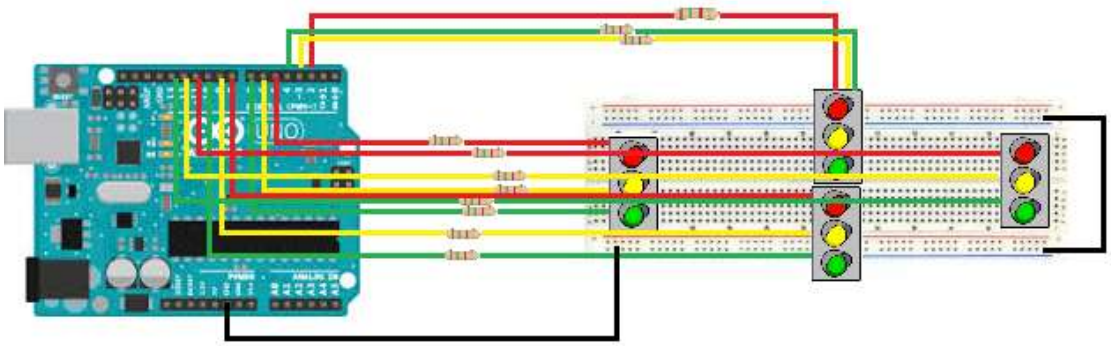

Figura 10. Esquema semáforo do Projeto

10.2 Telas e código fonte do Projeto

Finalizando na Figura 12, 13, está a aplicação criada para controlar e gerenciar as informações das câmeras. Nesta aplicação podemos usar filtros de visão computacional para conseguir um ajuste mais preciso das informações.

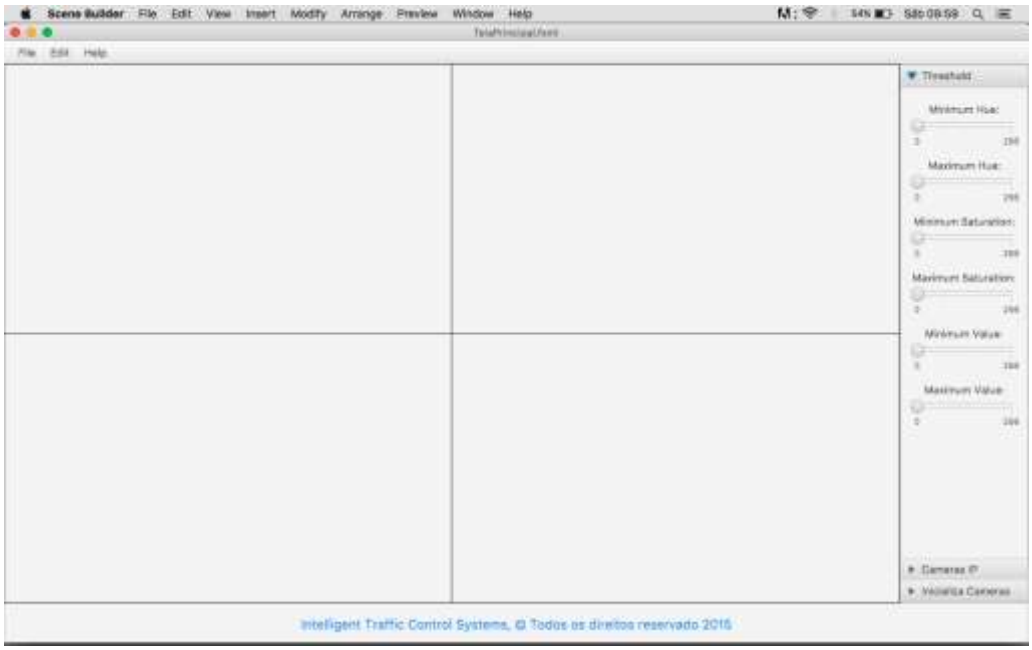

Figura 11. Esquema Final do Projeto

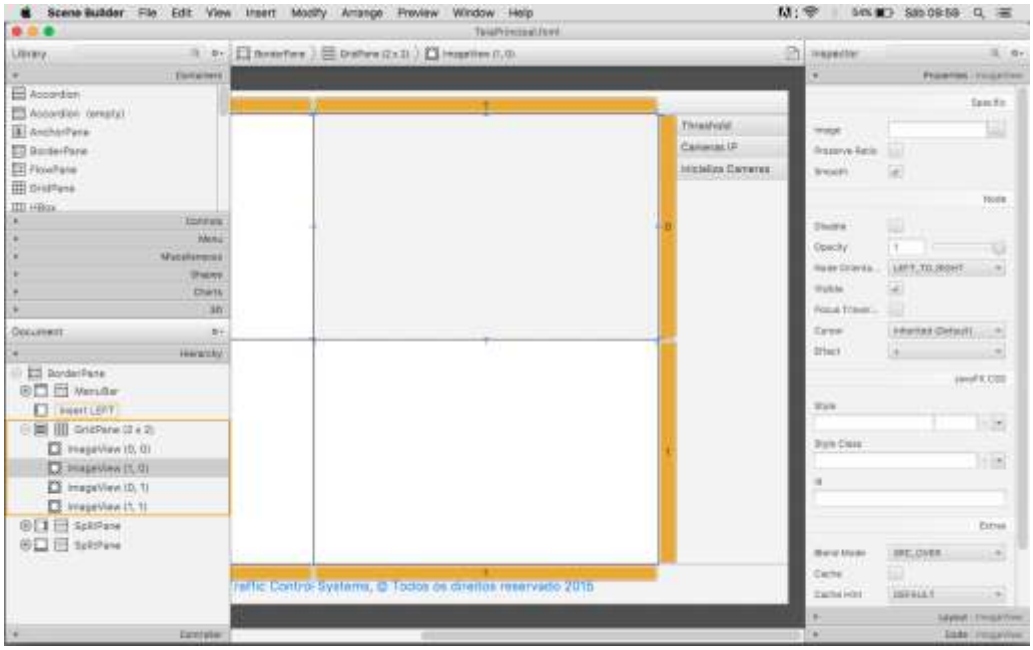

Figura 12. Esquema Final do Projeto

Finalizando na Figura 13, 14, está um trecho do código fonte da aplicação criada para controlar e gerenciar as informações das câmeras.

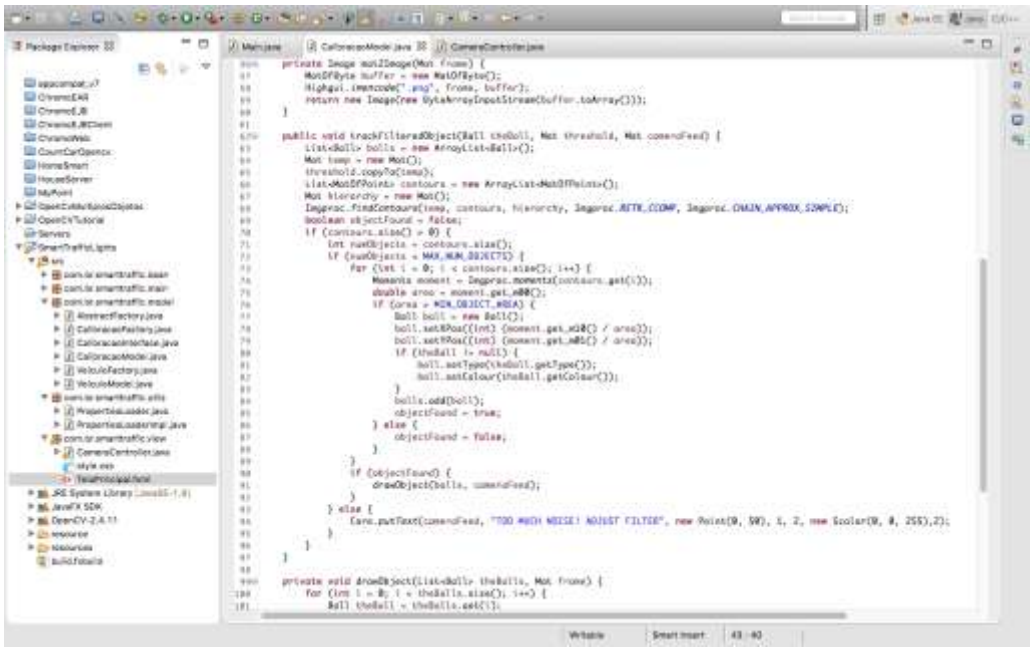

Figura 13. Código fonte do Projeto

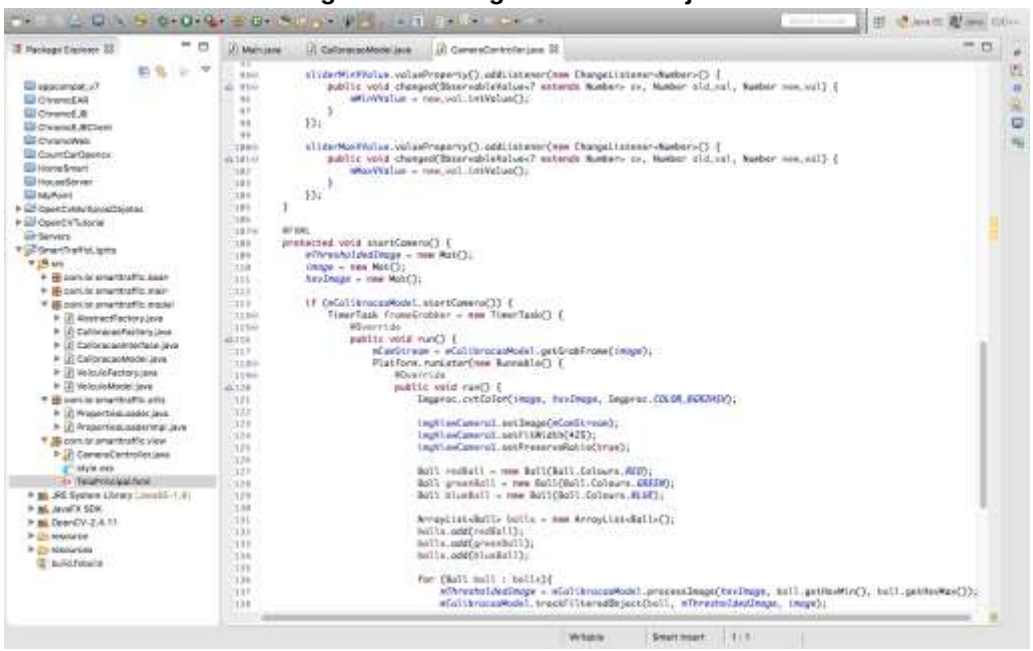

Figura 14. Código fonte do Projeto

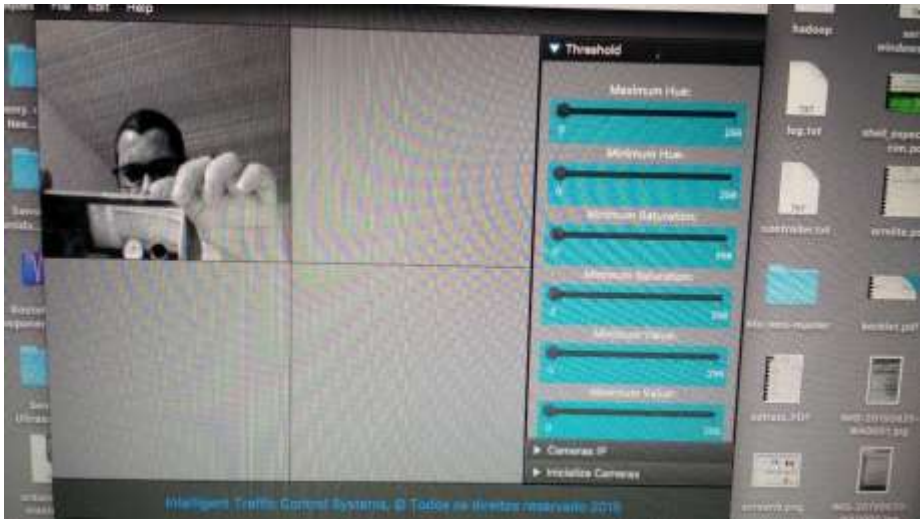

Figura 15. Aplicação usando Java FX e OpenCV

Na figura 15 temos uma demonstração da aplicação de detecção das câmeras. Conforme o fluxo de trânsito, pode se fazer o gerenciamento do tempo de abertura e fechamento dos semáforos, proporcionando melhores condições no trânsito e mantendo a harmonia entre os condutores, viabilizando o trânsito de forma organizada e controlada.

As câmeras também farão um papel muito importante, pois armazenarão informações dos dados dos veículos em circulação. Nos casos mais específicos, como controle de furtos, pode se pegar as características dos veículos em circulação, fazendo a leitura computacional, comparando as características do veículo furtado.

CONSIDERAÇÕES FINAIS

Neste trabalho foi proposto o desenvolvimento de um protótipo para o controle do fluxo de trânsito nas cidades, com o objetivo de melhorar a forma que estamos lidando com o caos no trânsito nas diversas metrópoles pelo mundo. O foco principal deste modelo é mostrar que com recursos tecnológicos simples, mas eficientes, podemos conseguir elaborar ideias que podem solucionar problemas que atuam no nosso dia-a-dia, e com isso incentivar a comunidade na elaboração de novos projetos, solucionando assim, os problemas que temos na infraestrutura da sociedade. Muitos recursos podem ser obtidos com o uso de câmeras, não podemos limitar somente o seu uso para monitoramento, mas usá-lo como uma fonte de dados.

Referências

Ballard D. H. (1982), Computer Vision, PrenticeHall.

Brahmbhatt S. (2012), Practical OpenCV, Apress, Tecnology in action

Dea C. (2010), JavaFX 2.0, introduction by example, apress

Deitel P. e Deitel H. (2010), Java com programar, 8 edição, Person

Decicino R. (2008) "Trânsito: Problemas atingem grandes cidades", http://educacao.uol.com.br/disciplinas/geografia/transito-problemas-atingem-grandes-cidades.htm

Evans B. (2011), Beginning Arduino programming, Apress, Tecnology in action

Evans M., Noble. J., Hochenbaum J., (2013), Arduino em ação, 1 edição, novatec.

Figura 1 disponível em: http://www.tecmundo.com.br/google/61734-google-mostra-sistema-reconhecimento-imagens-carros-autonomos.htm acesso em out/2015.

Figura 2 disponível em: http://www.technologyreview.com/view/535201/the-face-detection-algorithm-set-to-revolutionize-image-search/ acesso em set/2015.

Figura 3 disponível em: http://www.techwarelabs.com/ces-2012-microsoft/ acesso em set/2015.

Figura 4 disponível em: http://opencv-java-tutorials.readthedocs.org/en/latest/03%20-%20First%20JavaFX%20Application%20with%20OpenCV.html acesso em set/2015.

Figura 6 disponível em: http://www.cbpf.br/cat/pdsi/lpr/lpr.html acesso em set/2015.

Java e orientação a Objetos FJ 11, Caelum

Margolis M. (2011), Arduino CookBook, O'Reilly.

Maria M. P. (2009) "Trânsito de Maringá está à beira dos caos", http://www.gazetadopovo.com.br/vida-e-cidadania/maringa/transito-em-maringa-esta-a-beira-do-caos-bm71f8u7f9vremii1k0vw6z4e

McRoberts, M. (2011), Arduino Básico, Novatec.

SOUZA, C. F.; CAPOVILLA, F.; ELEOTÉRIO, T. C.(2010), Visão Computacional. Faculdade de Tecnologia de Taquaritinga, Centro Estadual De Educação Tecnológica "Paula Souza", Taquaritinga.

Timmis H. (2011), Practical Arduino Engineering, Apress, Tecnology in action

Oliveira B. (2014), JavaFX, Interfaces com qualidade de desktop, Casa do código.

Weaver J., Gao W., Chin S., Iverson D., Vos J. (2012) Pro JavaFX 2, "A definitive guide to rich clientes with java tecnology, apress.