Detecção de objetos em um jogo em tempo real

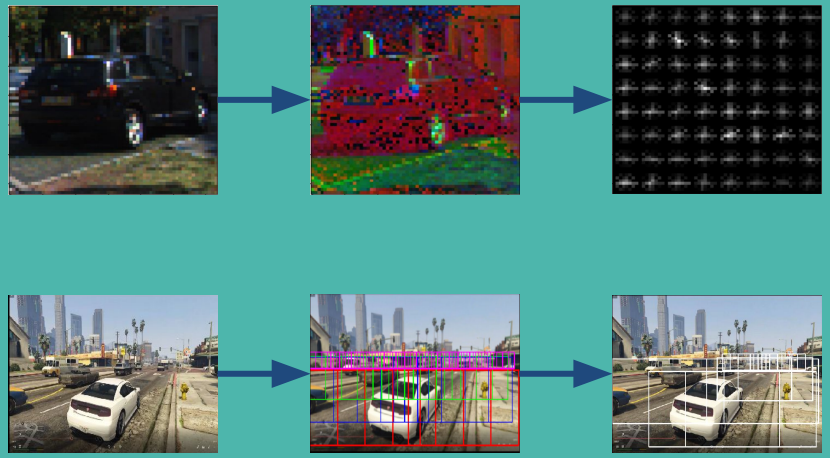
Robson Zagre Jr. e Nancy Baygorrea

Introdução

- Evolução dos Gráficos e Tecnologias em Video Games
- Ambiente controlado e conhecido
- Testes em simuladores em contraste com a realidade
- Visão Clássica e por Deep Learning
- Necessidade de performance para tempo real

Clássica

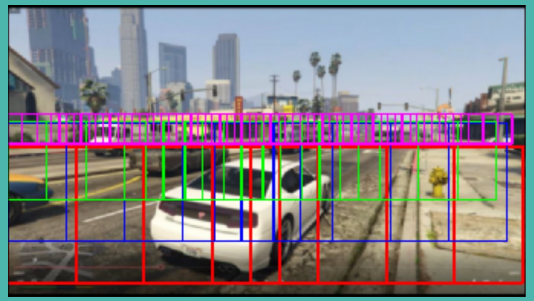
- Utilização de trabalho base (changsub)
- Dataset com imagens reais de veículos e não-veículos
- Pipeline Imagem:
  - HSL
  - Histograma
  - HOG
- Pipeline Window
  - Resize para 480x640
  - Filtro Gaussiano
  - Definição de Janelas



Clássica

- Utilização de SGD Classifier por permitir treinamento online
- Desenvolvimento de um CustomSGDClassifier
- Incorporação do pipeline internamente
- Grid Search nos parâmetros de pré-processamento em 50 imagens
- Pode ser melhorado com pós-processamento (Heatmap)
- Aproximadamente 1 frame por 1.3 segundos, em contraste com 1 frame por 4.28 segundos.

```python
def search_for_good_pipe_params(X_train, y_train):
    print("Search for Best Params")
    # Use a custom model to define best params used in pipeline of feature generate for model
    parameters = dict(
        color_type=['HSL','HSV'],
        orient=[7,8],
        pix_per_cell=[4,8],
        cell_per_block=[2,4,6],
        block_norm=['L1','L2'],
        transform_sqrt=[False,True],
        bins=[16,32]
    )
    model = CustomSGDClassifier()
    modelgscv = GridSearchCV(model, parameters, n_jobs=-1, refit=True, return_train_score=True)
    #Look just for a part because of computer complexit and RAM
    modelgscv.fit(X_train.iloc[0:50], y_train.iloc[0:50])
    with open(params_path, 'w', encoding='utf-8') as f:
        json.dump(modelgscv.best_params_, f, ensure_ascii=False, indent=4)
    return modelgscv.best_params_
```



Deep Learning

- YOLO v3
- 80 classes
- YOLOv3-320
  - Maior número de detecção
  - Menor performance
- YOLOv3-tiny
  - Menor acurácia
  - Maior a performance

Observações

- GTA V (qualidade gráfica próxima a realidade).
- Gráficos em modo normal e resolução de 800x600
- Core i5-7300HQ
- 8GB de RAM
- GeForce GTX 1050

Conclusão

- Abordagem clássica encontra dificuldades em generalização
- Resultados podem ser melhorados com pós-processamento
- Performance aceitável com 1 frame por 1.3 secs.
- Utilização de Grid Search para encontrar os melhores parâmetros (Pipeline)
- YOLO v3 demonstrou boa generalização e desempenho
- Avanços tecnológicos permitem simulações em ambientes controlados e conhecidos.

Referências

- https://github.com/windowsub0406/Vehicle-Detection-YOLO-ver
- https://www.youtube.com/watch?v=GGeF_3QOHGE&list=PLMoSUbG1Q_r8nz4C5Yvd17KaXy8p0ufPH