This Research Unit is funded by National Funds through the FCT - Foundation for Science and Technology, in the context of the project UID/CEC/00127/2019.

Detection of Data Matrix Encoded Landmarks in Unstructured Environments using Deep Learning

Authors:

Tiago Almeida, DEM, IEETA, University of Aveiro, Portugal

Vítor Santos, DEM, IEETA, University of Aveiro, Portugal

Bernardo Lourenço, DEM, University of Aveiro, Portugal

Pedro Fonseca, IT, DETI, University of Aveiro, Portugal

15/04/2020

Presentation Outline

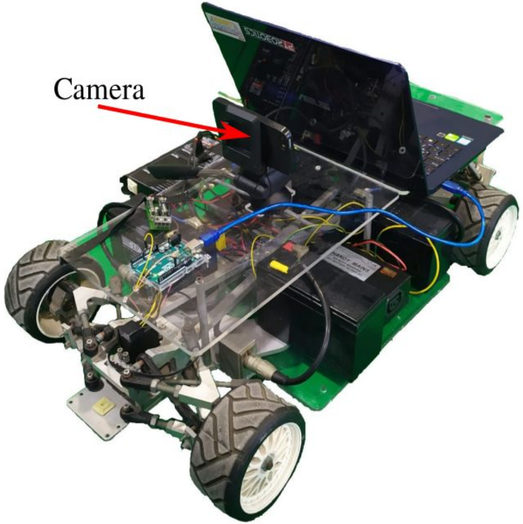
1. Introduction

Produtech Project

- Development of a flexible and low-cost localization and navigation system
- Mobile robot capable of computing its own location in real-time
- Robust system to perform in complex and unstructured environments such as production facilities
- Detect encoded landmarks (with its own location) and then apply triangulation/trilateration techniques

1. Introduction

How do we do that?



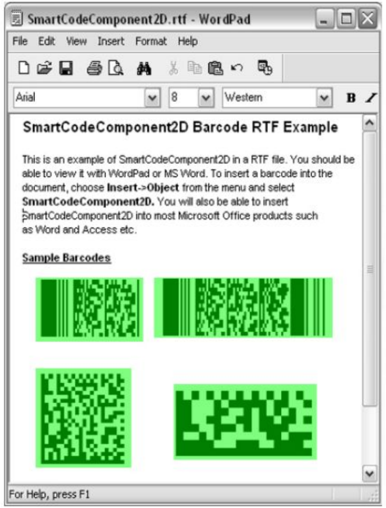
2. Related Work

2. Related Work

Object Detection DL Approaches

- Faster RCNN
- YOLO (all versions)
- SSD (Single Shot Detection)

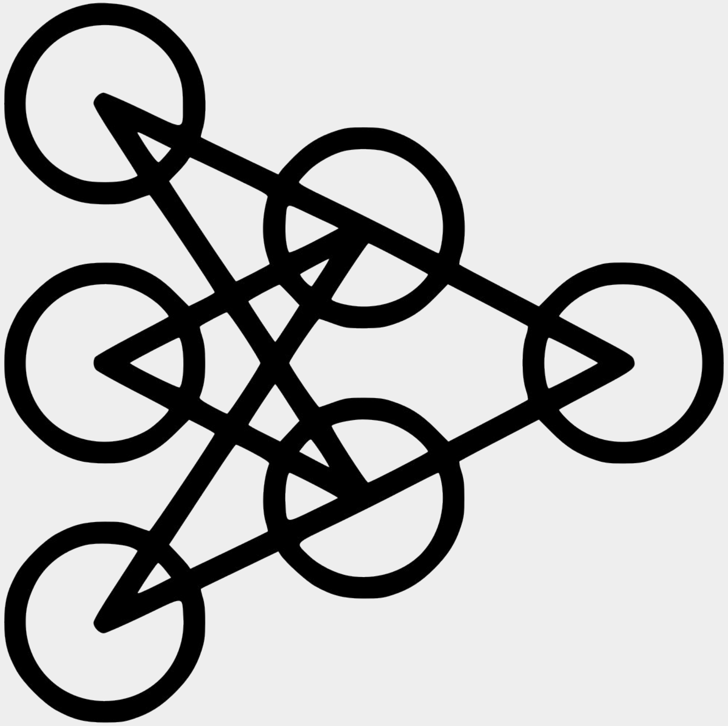
Data Matrix Detection

- 

  from: A.Zharkov and I. Zagaynov, “Universal barcode detector via semantic segmentation”, Available: http://arxiv.org/abs/1906.06281

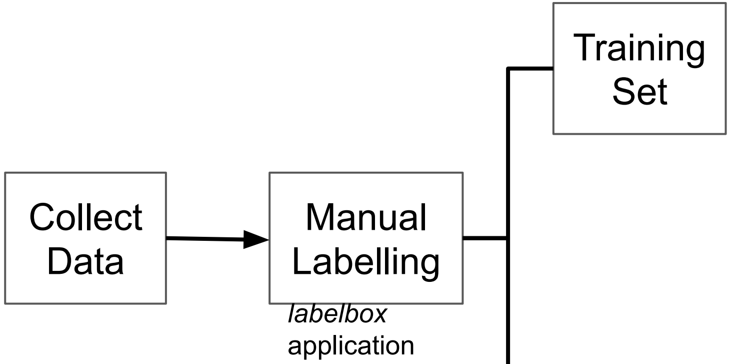
- Classic algorithm from libdmtx (Python library)

3. Proposed Approach



3. Proposed Approach

a. A dataset of Data Matrix Images



Lab (78 frames)

Workshop I (78 frames)

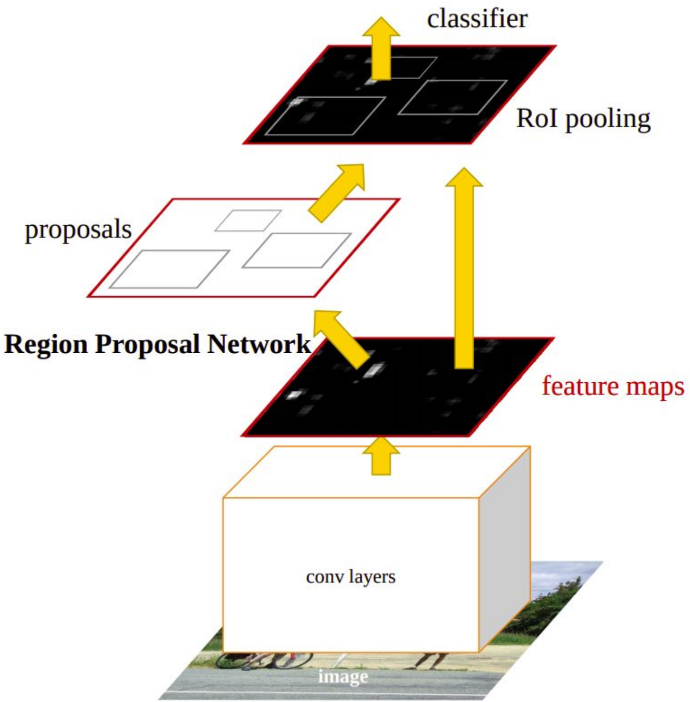
Hallway (158 frames)

Workshop II (66 frames)

3. Proposed Approach

b. Faster RCNN Architecture

- 2 modules(deep fully convolutional network and classifier)
- Region Proposal Network (RPN)
- Faster R-CNN detector
- Non-Maximum Suppression (NMS)



3. Proposed Approach

c. DNN Training Details

- Detectron2 research platform (Pytorch)
- Training images size: 8000×6000
- Data Augmentation: image resizing and random flip
- Learning rate: 0.00025 during 4000 iterations
- NVidia RTX2080ti GPU
- NMS threshold: 0.7

4. Experiments and Results

4. Experiments and Results

Evaluate the trained model on the test set

- COCO dataset metrics:
  - Average Precision (AP): $$P = \frac{T_P}{T_P + F_P}$$
  - Average Recall (AR): $$R = \frac{T_P}{T_P + F_N}$$

T_P (True Positive)

F_N (False Positive)

4. Experiments and Results

Experiments and Results

AR>AP

- Both metrics averaged over multiple IoU thresholds (threshold $\epsilon[0.50; 0.95]$ in steps of 0.05 $\longrightarrow [0.50 : 0.05 : 0.95]$)

- For AP, a constant IoU thresholds of 0.5 ($AP@0.5$), 0.75 and 0.95 are also computed

- Small objects: $area < 32^2$, medium objects: $32^2 < area < 92^2$ and large objects: $area > 92^2$

- For AR, maximum number of detections ($maxDets \ \epsilon\{1, 10, 100\}$)

	IoU Thresholds	Scales	maxDets	AP/AR values
AP	[0.50 : 0.05 : 0.95]	all	100	0.619
	0.50	all	100	0.876
	0.75	all	100	0.730
	0.95	small	100	0.364
	[0.50 : 0.05 : 0.95]	medium	100	0.565
	[0.50 : 0.05 : 0.95]	large	100	0.737
AR	[0.50 : 0.05 : 0.95]	all	1	0.288
	[0.50 : 0.05 : 0.95]	all	10	0.669
	[0.50 : 0.05 : 0.95]	all	100	0.670
	[0.50 : 0.05 : 0.95]	small	100	0.438
	[0.50 : 0.05 : 0.95]	medium	100	0.611
	[0.50 : 0.05 : 0.95]	large	100	0.786

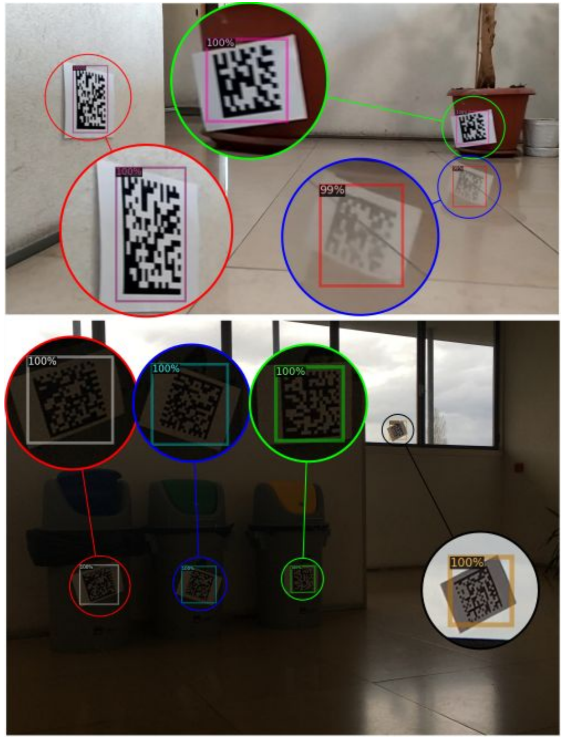
4. Experiments and Results

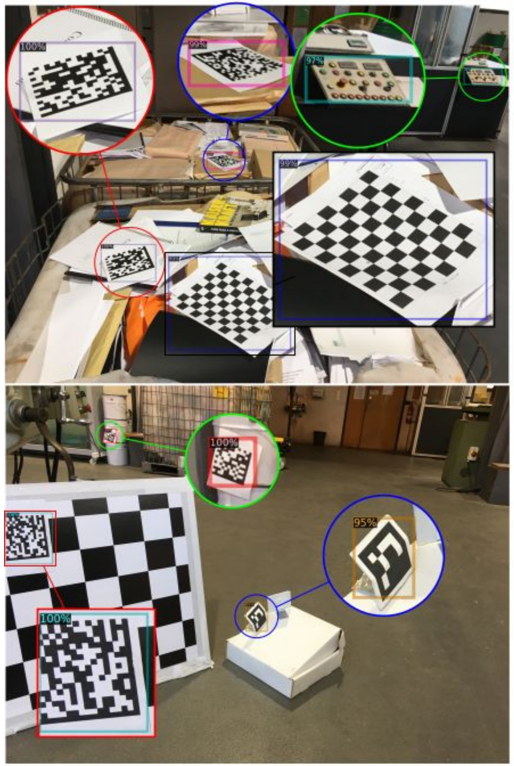
Running Time Results and Comparisons

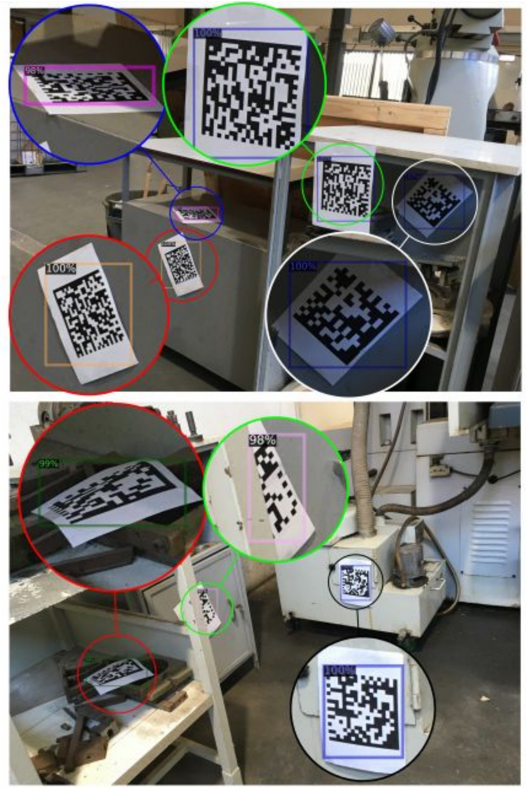
- 144 ms per image (7 fps)
- Only 45% of the test set frames were accurately processed by the classic algorithm
- The running time for the classic algorithm was 4.97 s per image (40 times slower than the model that we use in this paper)

4. Experiments and Results

Qualitative Results







5. Conclusions and Future Work

5. Conclusion and Future Work

- The model performs accurately and it is consistent by detecting almost all the landmarks in the test set;
- Overcomes by far, in performance and frame rate, the traditional algorithms;
- This implies a self-localization system more robust and performant;
- Future work; the development of a DNN whose outputs are parallelograms instead of simple rectangular bounding boxes; this would provide a novel technique for finding the transformation robot-marker with only one necessary landmark.

https://github.com/tmralmeida/faster-rcnn-data-matrix

15/04/2020