
# Mini-Project Report

# Topics in image processing

**Mini-Project in topics of image processing 202-1-4902**

**Ido Alkalai Tavori 209260017 idoalk@post.bgu.ac.il**

**Noa Cohen 316076066 noaap@post.bgu.ac.il**

**Supervisor: Prof. Jihad El sana**

# 1 Introduction

The topic of our project is "Track & Draw, maintain the draw over the object". In this project we were required to explore tracking libraries and algorithms and compare 3 of them. We learned how to use each tracking library and algorithm, compared them against each other (cons and pros) and evaluated their usefulness. We drew over the tracked object with each library, and observed the results.

# 2 Methods

The three libraries we studied were: Vuforia, MOSSE (adaptive correlation filters) and medianflow.

## 2.1 Vuforia:

**Requirements:**
**- Model Target Generator program**
**- Unity engine**
**- Vuforia library**

Vuforia is an augmented reality library that includes an object tracking feature. It can be used with either C++ or the Unity engine. We have chosen to use Unity.
We have than followed these steps:
**1** Download a 3d model (obj, cad…) that resembles the object we wanted to track as closely as possible.
**2** We uploaded the 3d model to the Model Target Generator
**3** Using the 3d Model Target Generator we configured the 3d model to suit the dimensions of the real object as closely as possible.
**4** Finally we generated the model target.
**5** *optional* Train the algorithm to recognize the model target by pressing the 'train' button in the 3d Model Target Generator. Not doing this step may result in less accurate tracking, but it will still function.
**6** We create a new Unity project, import the Vuforia library into it, and add an AR camera into the project (inserting the Vuforia activation key in the process).
**7** We add a model target to the AR camera: We set the database to be the generated model target we created in step 4.
**8** We run the project: We place the object at the correct distance for the tracking to work. The tracker should than project the 3d object into the real object.

## 2.2 MOSSE and medianflow

**Requirements:** - python
**-**opencv
**-**YOLOv3 weights
Packages: -imutils.video
-argparse
-imutils
-time
-cv2

The tracker is implemented using python, YOLOv3 weights and opencv.
**1** open Object_Tracking-main folder in cmd

**2** Run either these two commands (according to which tracker you wish to use):

**python opencv_object_tracker.py --tracker  mosse**

**python opencv_object_tracker.py --tracker  medianflow**

**3** The program will activate the computer's camera. We than press 's' to pause the recording, and choose the object to track – we do this by dragging the mouse over the picture and selecting the object by enclosing it in a rectangle.

**4** Once we have chosen the object, we press 'space'

**5** The program than resumes recording, while drawing a rectangle around the chosen object, according to the tracking algorithm chosen.

**6** Finally, we press 'q' to terminate the program.

# 3 Comparison

**Vuforia:**

**Pros:** - Very 'sophisticated' tracking compared to the others: draws a 3d object, supports more advanced drawing easily (like animation, and depth).

-Tracks objects accurately even when occulted or out of screen

- Very accurate tracking – draws precisely over tracked object
- Tracking is fairly fast and efficient.
- Coming hand in hand with the entire Vuforia library gives it very great AR capabilities that the other algorithms lack.

**Cons:**
- Very expensive: requires both a detailed model of the object to be made and to buy the Vuforia library (which is quite costly).

- Needs to be trained in order to run optimally.

- Does not track moving object well: can track small movements, but loses object when it moves through bigger distances.

- Cannot track objects with moving parts within them.

- Fairly difficult to use by non programmers (compared to the other trackers discussed)

- Requires knowledge of Unity to use properly.

**MOSSE:**

**Pros:**

- Robust to variations in lighting, scale, pose, and non-rigid deformations

- Able to detect occlusion, pause and resume tracking when the object returns to the frame.

- Operates at high fps

- Fairly easy to implement

- Relatively faster tracking compared to other algorithms.

**Cons:**
- Lags behind deep learning based trackers in terms of performance

- Not very sophisticated tracking (unlike Vuforia) – can draw an image

over an object, but that's it.
- Not a lot of built in robust AR functions.

- Not as accurate as Vuforia.

**Medianflow**

**Pros:** -Works well will predictable and small motions

- Recognizes when tracking has failed and stops when it happens (unlike other trackers, which can result in haphazard and incredibly inaccurate tracking)

- Relatively simple to implement

- Very easy and simple to use

- Can track moving object, unlike Vuforia

- moving parts within object don't necessarily result in tracking failure

**Cons: -** Fails under larger motion

- Fails when object is occulted

- Tracking is not very sophisticated (only draws image over object, unlike Vuforia)

- No built in AR support

- Not as fast as MOSSE

- Tracking not as accurate as Vuforia.

# 4 Conclusions

After implementing, using and observing each of the 3 tracking algorithms, we have come to the conclusion that:

- Medianflow, although simple and has a good failure reporting, is a bit unimpressive in comparison to the other trackers, and we would recommend MOSSE over it for simple model tracking.

- MOSSE is relatively very fast, and quite accurate, but its inability to continue tracking after occultation leaves it at a disadvantage in comparison to Vuforia. On the other hand, it is very simple to both code, implement and use, thus we would still recommend using it for simple object tracking. It also much better than Vuforia for tracking a moving object, and we would recommend using it for those purposes.

- Vuforia is undoubtably the most impressive of the 3, for its very accurate tracking capabilities and AR support. We would recommend it for creating VR tutorials as they are comfortable to code using the library and its tracking through occultation and 3d model drawings make it perfect for this purpose. However, its expensiveness makes it hard to recommend for non professional or for low budget programmers, and definitely not to non programmers.

## 5 Bibliography

https://learnopencv.com/object-tracking-/using-opencv-cpp-python

https://www.curiscope.com/blogs/blog/what-is-an-augmented-reality-tracker-and-how-do-i-use-one

https://codereality.net/ar-for-eu-/book/chapter/foundations/hardware/tracking

https://library.vuforia.com/features/environments/device-tracker-overview.html