COMP9517 Project Report

- **Author: Yunqiu Xu (z5096489)**

---

1. Introduction

The aim of this project is to detect and track multiple objects in a video. The program should be both accurate and fast enough to achieve real-time tracking. Additionally, some special cases should also be considered and handled including occulusion, out-of-scene object and similar objects. The source code as well as sample videos are available on https://github.com/YunqiuXu/MultiObjMatching

2. Method

The work flow is shown in Figure 1:

- In step (1), the first frame is treated as model frame and bounding box as well as contour for each object are drawn manually.
- To speed up the process, both the model frame (2) and new frames (4) are transformed to lower resolution images using Gaussian Pyramide
- In this project tracking is performed in ahead of matching. Several kinds of discriminative filters including Kernalized Correlation Filter(KCF), Median-flow Filter and Tracking Learning Detection (TLD) are initialized in (3) and used to track bounding boxes in (5). Compared with generative filters such as Kalman Filter, these filters regard tracking as classification problem to classify object from background. For each frame discrimative features can be extracted to train the classifier to make prediction in next frame. In this way we can get the ROIs before matching.
- After tracking, the bounding boxes in 2 images are cropped (6) to perform matching (7). SURF and ORB are implemented to generate descriptors and matches are collected using FLANN. These match pairs need to be refined to

compute homography matrix, then the contours on model frame are mapped to new frame. If there are only a few match pairs, it will be inaccurate to compute homography matrix, in this case all of the contours in each bounding box will be computed via Otsu threshold and the largest one will be treated as the contour for target object.

- Finally the result of tracking and matching will be remapped to original size to form matched image (8).

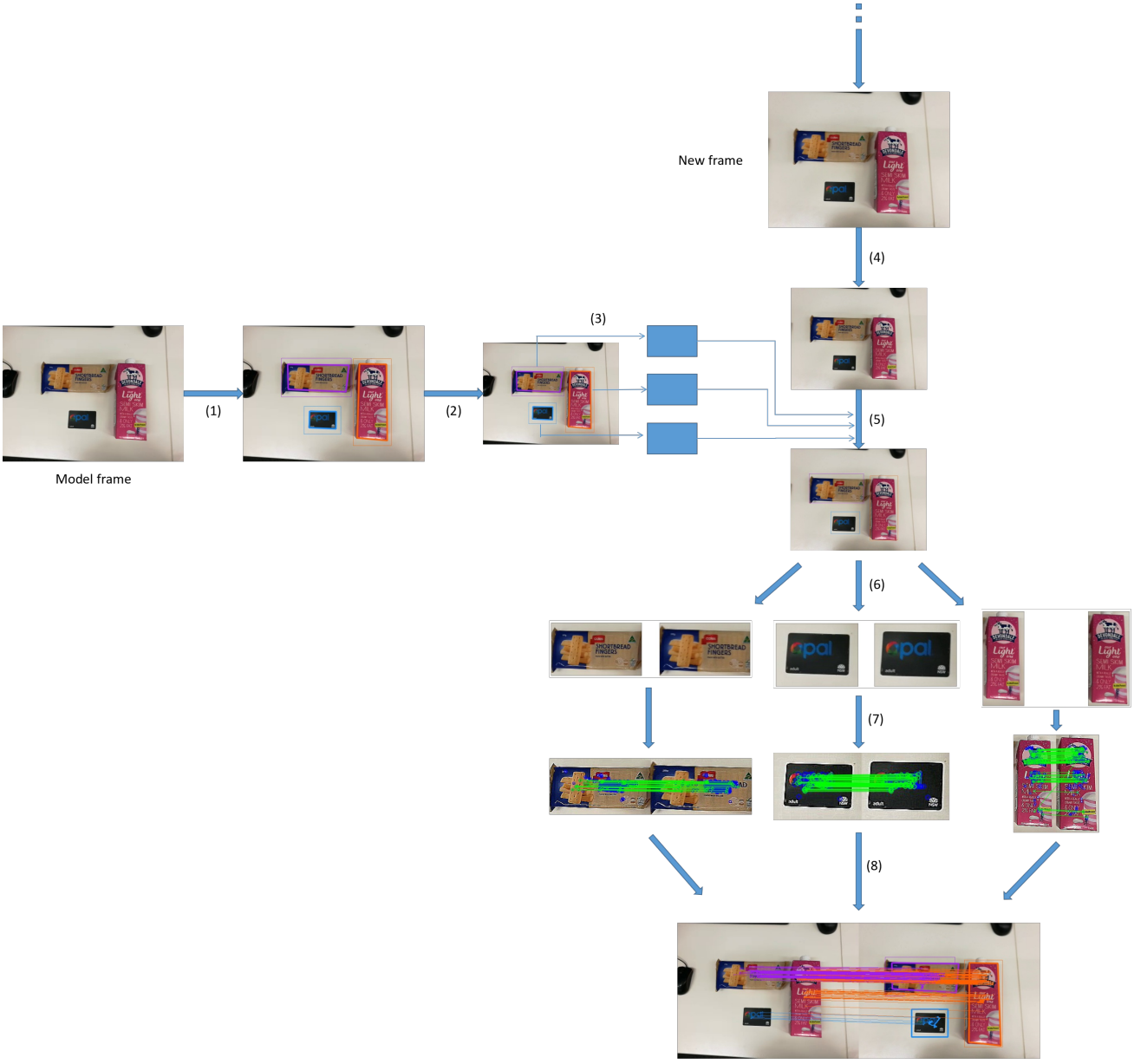

Figure 1 Work Flow

3. Implementation details

- The program is built with Python 2.7.11, dependencies : OpenCV 3.3.0, numpy 1.11.3, Pillow 3.1.2, numba 0.34.0. The running environment is Ubuntu 16.04 LTS with 4 Intel(R) Core(TM) i7-5600U CPU and 8 GB RAM
- In order to test the accuracy, efficiency and special case handle ability, there are 3 versions being tested:
  - V1: Both tracking and matching are performed in low resolution
  - V2: Tracking in low resolution and matching in original resolution
  - V3: Both tracking and matching are performed in original resolution

4. Result and Discussion

- Result shows that V2 achieves best performance among 3 versions that it can perform tracking and matching accurately with decent speed (FPS 20-30 for 2-3 objects). Matching is not performed well on low resolution images even when they are sharpened, especially when ORB is used, while tracking is not sensitive to resolution. For images with original size, although the matching can be very accurate with a lot of matched pairs, the speed can not meet the goal of real-time tracking.
- The selection of trackers will affect bounding box prediction, following are their preformance in different conditions. Compared with OpenCV built-in KCF, KCF with HOG shows more accurate tracking result but the performance is poor when object goes out of scene. TLD filter can achieve long term tracking however the general speed is very slow.

Tracker	General Speed	General Accuracy	Occulusion	Out-of-scene	Similar Objects
OpenCV KCF(without HOG)	Fast	Good	Short Term	Short Term	Good
Modified KCF(with HOG)	Fast	Good	Poor	Short Term	Good
OpenCV Median Flow	Very Fast	Median	Poor	Poor	Median

Tracker	General Speed	General Accuracy	Occulusion	Out-of-scene	Similar Objects
OpenCV TLD	Slow	Median	Long Term	Long Term	Median

- ORB can be a suitable alternative to SURF on original-resolution images that it can perform enough match pairs with 10 times faster speed. Another version is OK as well that perform both tracking and matching in low resolution but use SURF to generate descriptors, this method is slightly slower with similar accuracy.

5. Conlcusion and Outlook

In this project a real-time tracking-and-matching system is built to handle multiple objects in different conditions. It can make match accurately with a rather fast speed. Additionally, some improvements can also be achieved in the future:

- End-to-end detection: Currently we have to draw bboxes and contours on the first frame manually. By using some object detection methods such as Fast RCNN and YOLO we can locate them automatically at the very beginning.
- Some faster and more accurate trackers such as C-COT and ECO can be applied to improve the performance. Instead of built-in TLD, OpenTLD can be used for long term tracking to achieve better performance