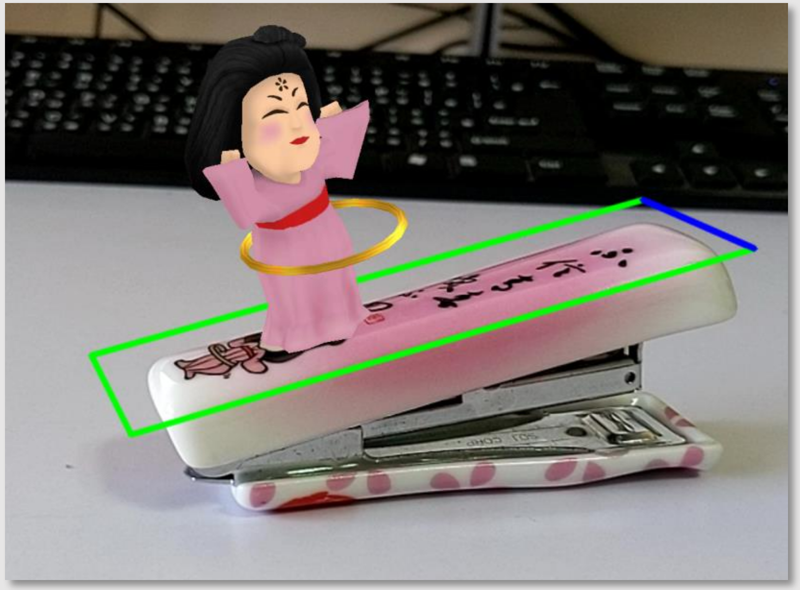
25th INTERNATIONAL CONFERENCE ON PATTERN RECOGNITION
ICPR 2020

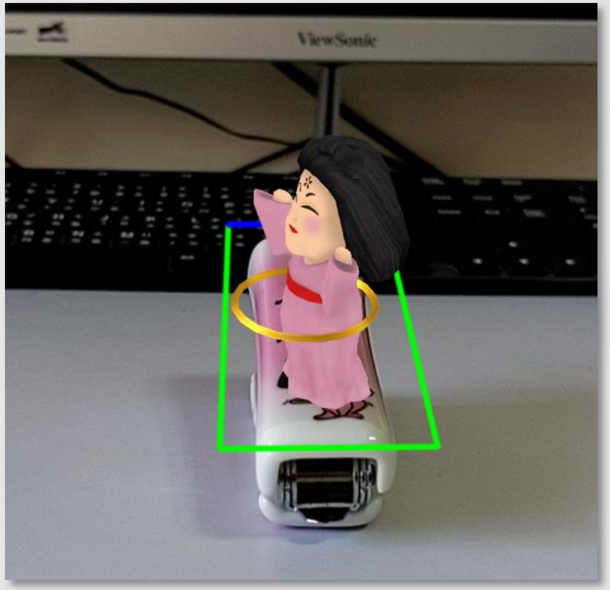
Mobile Augmented Reality: Fast, Precise, and Smooth Planar Object Tracking

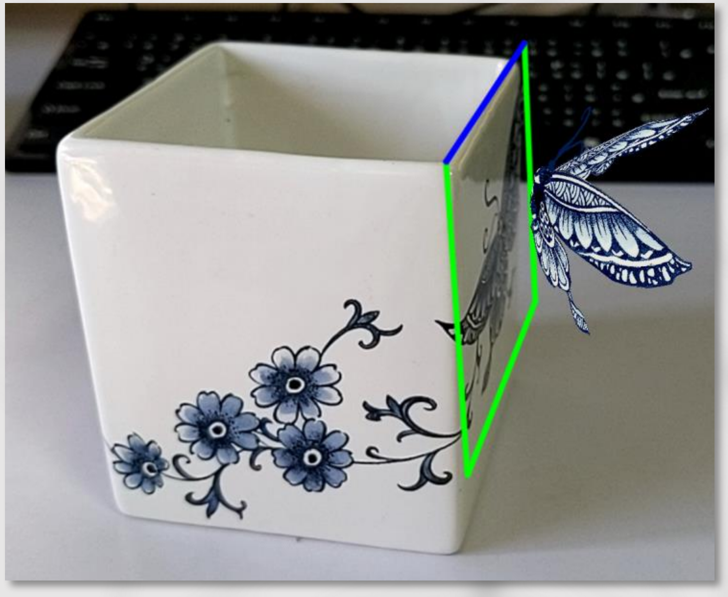
Dmitrii Matveichev[1] (xapocmat@gmail.com), Daw-Tung Lin[1]* (dalton@mail.ntpu.edu.tw)
Department of Computer Science and Information Engineering, National Taipei University
國立臺北大學 National Taipei University

Introduction

This work is focused on planar object tracking (POT) problem and it’s application in mobile augmented reality (AR).







The POT problem formulation

The POT problem can be formulated as follows:

- find the precise planar object position on a sequence of video frames using known object image.

POT Problems

The proposed algorithm solves following five common POT problems:

- **Extreme perspective transformation.**
- **Large scale-transformation**
- **Spatial jitter**
- **Degradation of tracking points number**
- **Optical Flow points drifting**

Optical flow with Binary Descriptors (OBD)

d_t: object descriptors

p_t: 2D object points

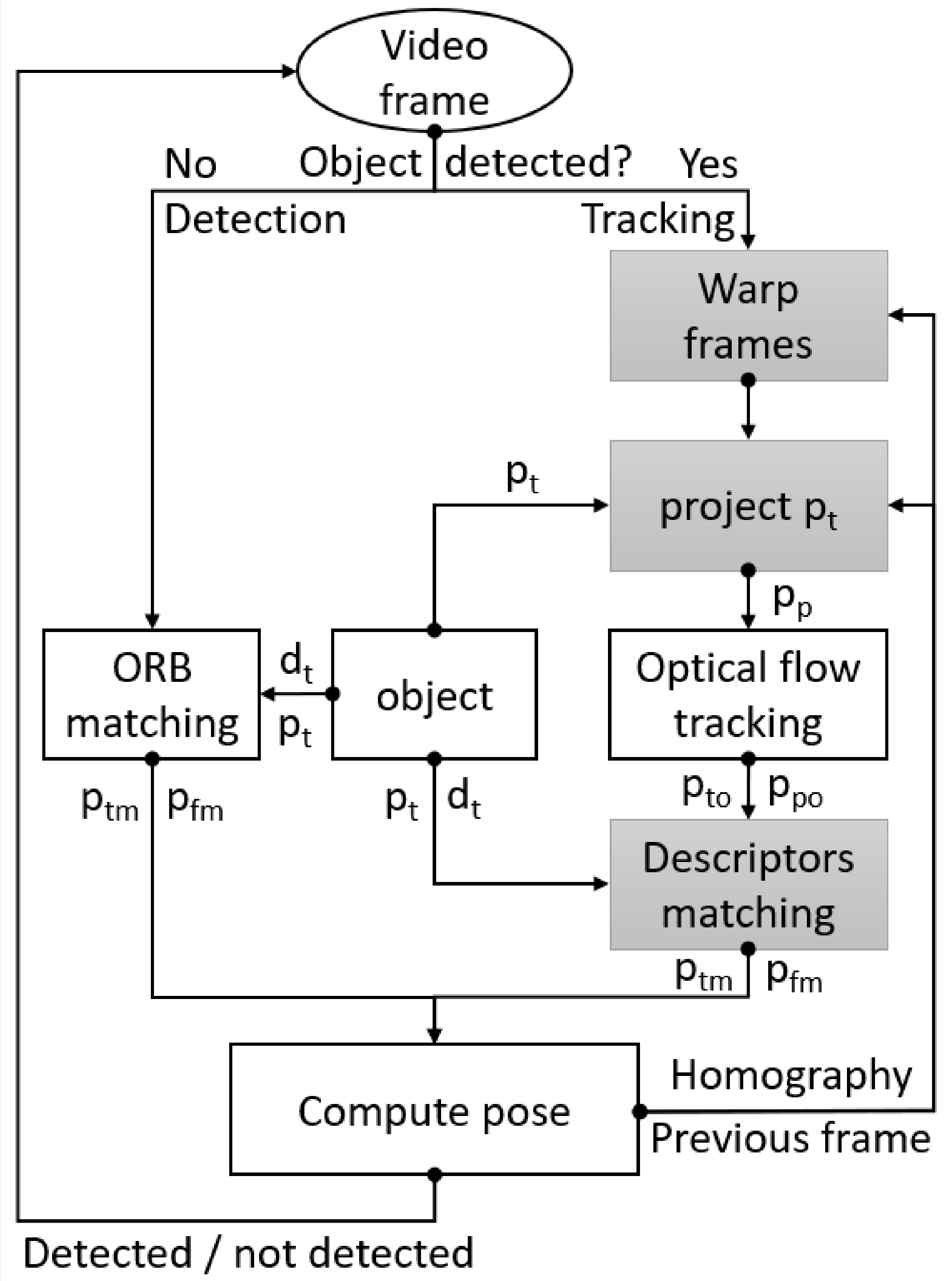
p_{tm}: matched object points

p_{fm}: matched frame points

p_p: projected object points

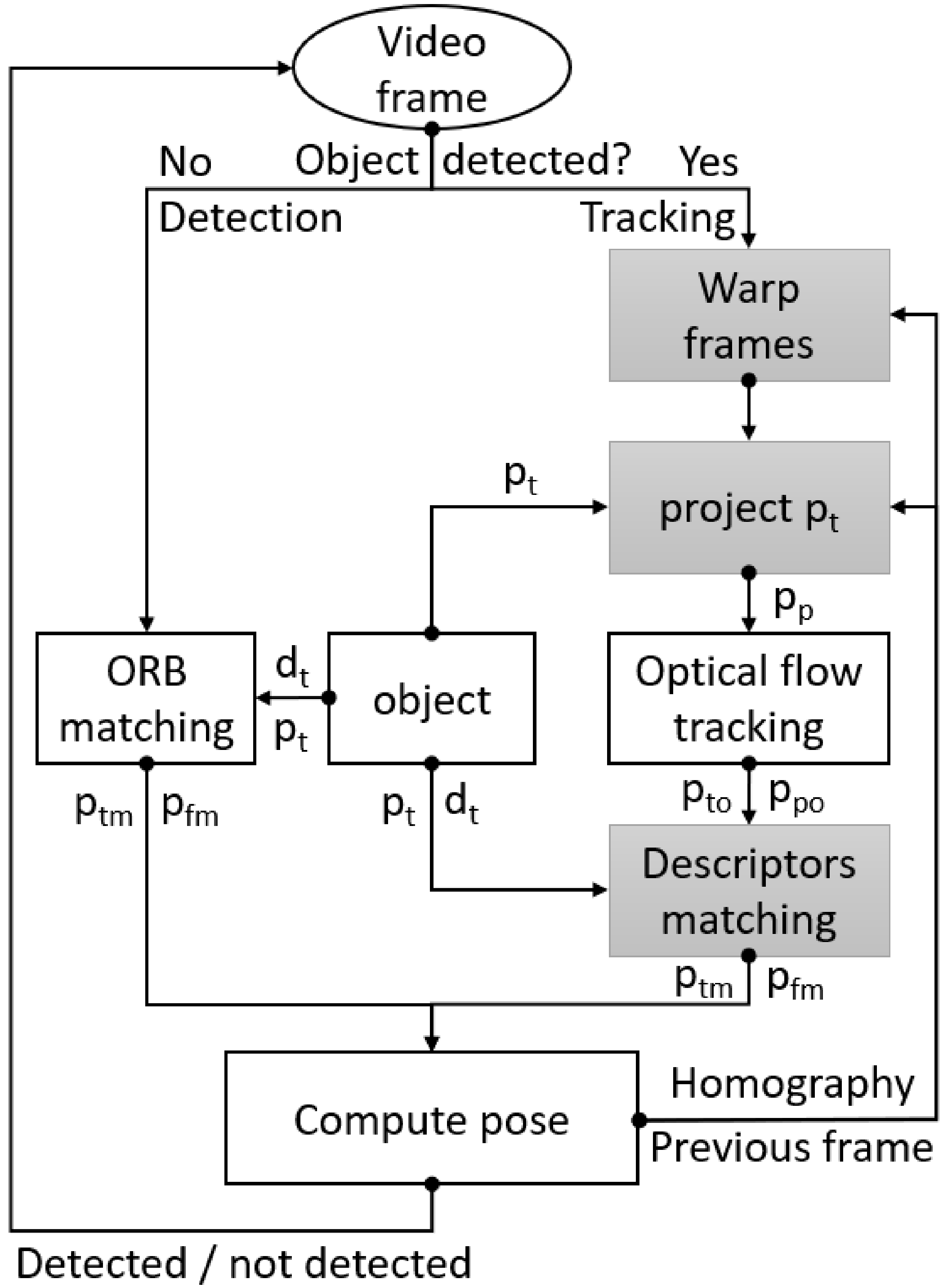
p_{po}: tracked projected object points

p_{to}: tracked target points



Detection Phase

- Detection is done by means of ORB binary descriptors detection and matching.
- The pose is computed based on matched points we
- output:
  - Homography
  - frame image
  - detection result



Tracking Phase

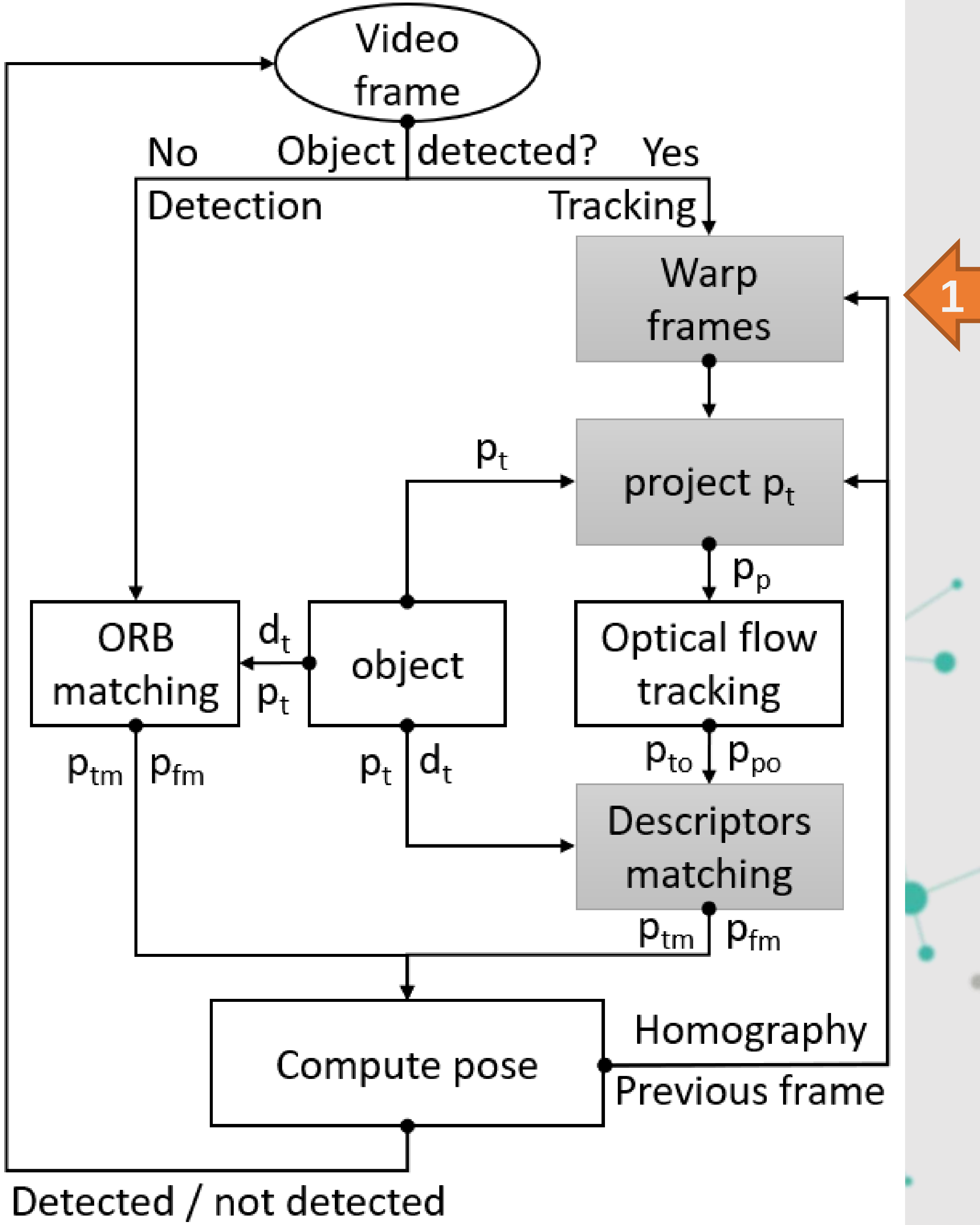
Starts if the object was detected in the previous frame.

For every frame performs:

1. Warp current and previous frames

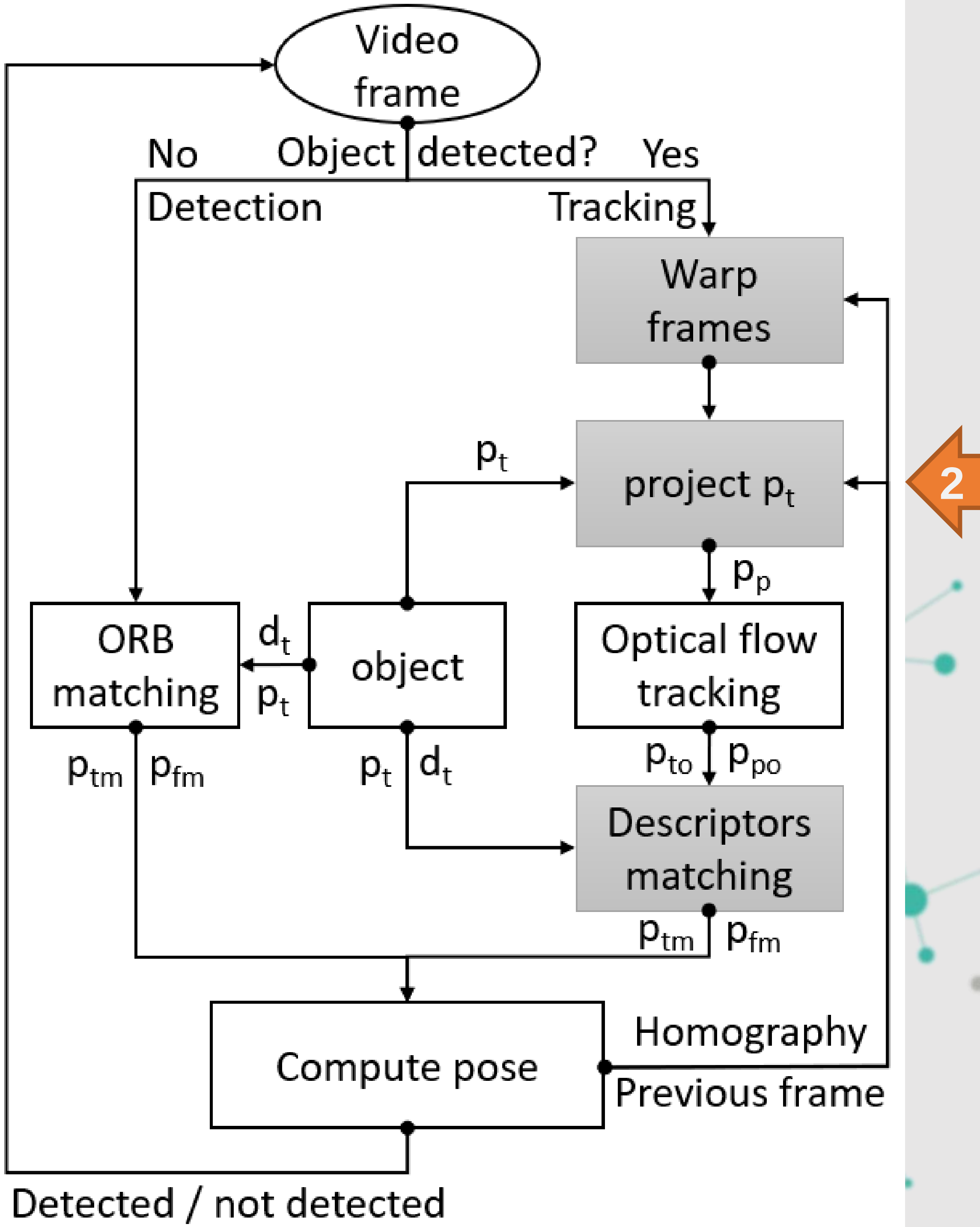
Warped frame

Warped previous frame



Tracking Phase

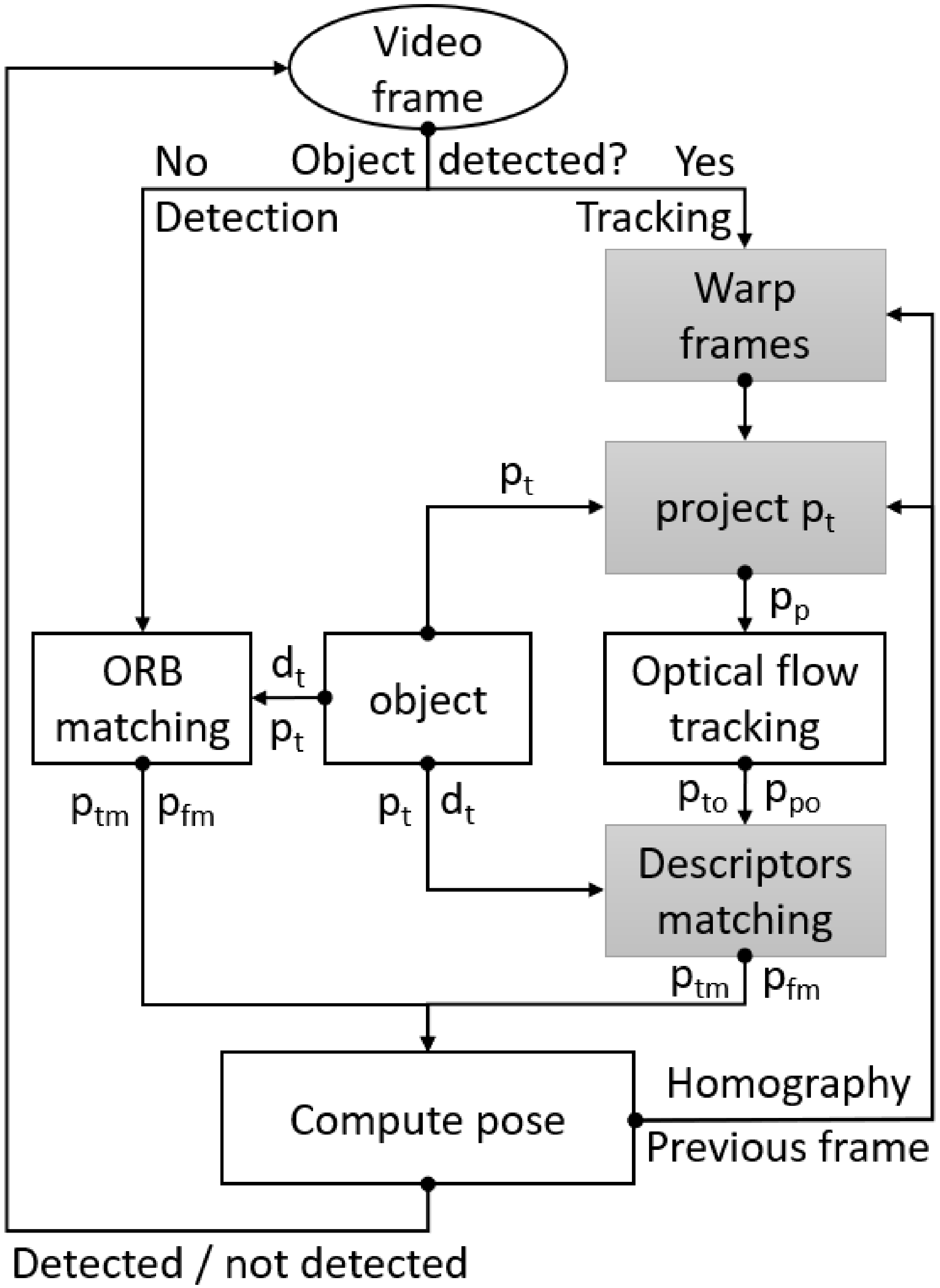
2. Using homography matrix, project the object points (p_t) to the current warped frame (points p_p);



National Taipei University

Tracking Phase

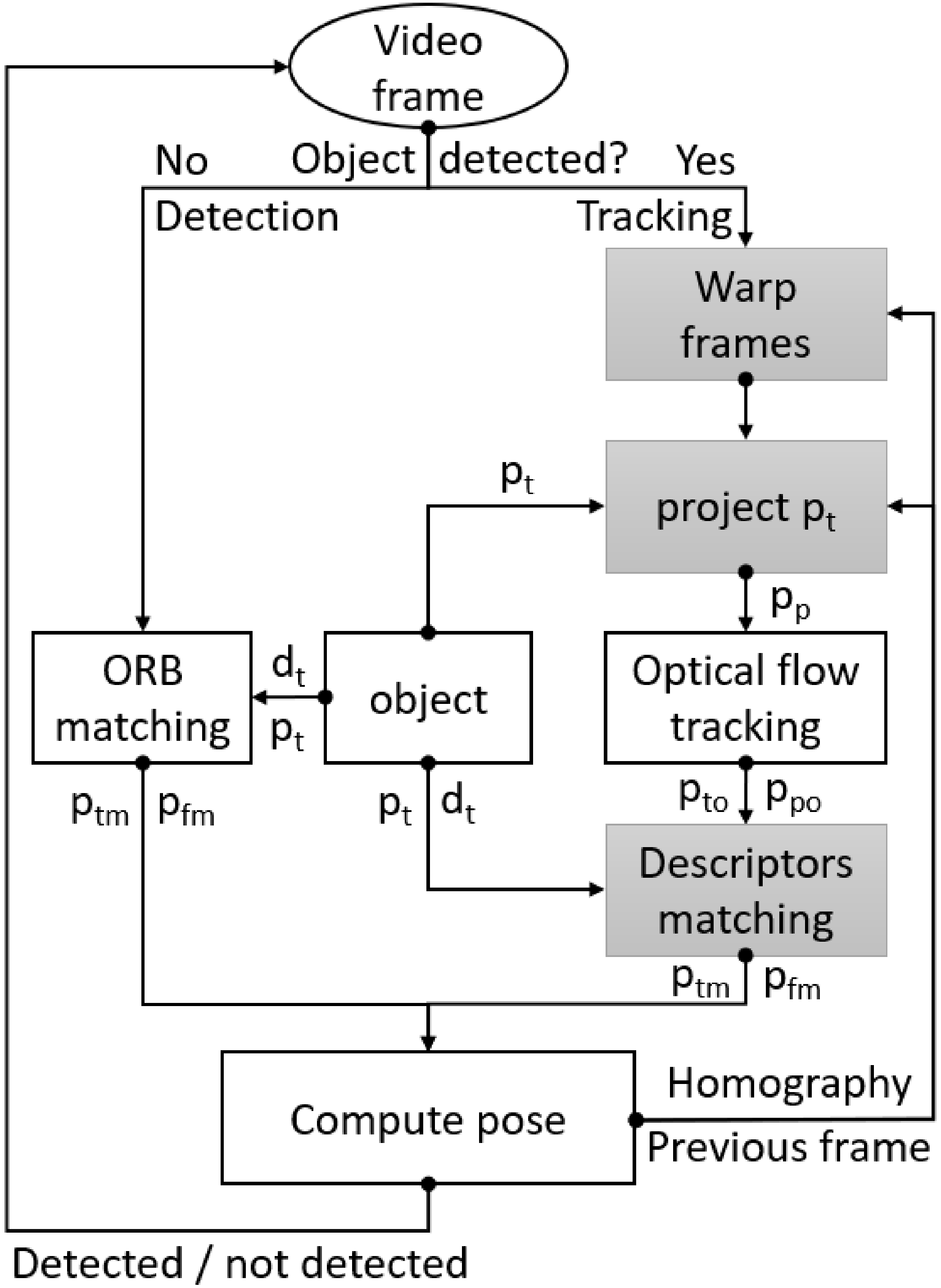
3. Do sparse OF points tracking with following input:
   - warped previous frame image
   - warped current frame image
   - projected object points (p_p);



National Taipei University

Tracking Phase

4. Filter the tracked points (p_{to} and p_{po}) using the descriptors matching based approach
   &
   Project matched frame point (p_{fm}) back to the current frame



4

國立臺北大學
National Taipei University

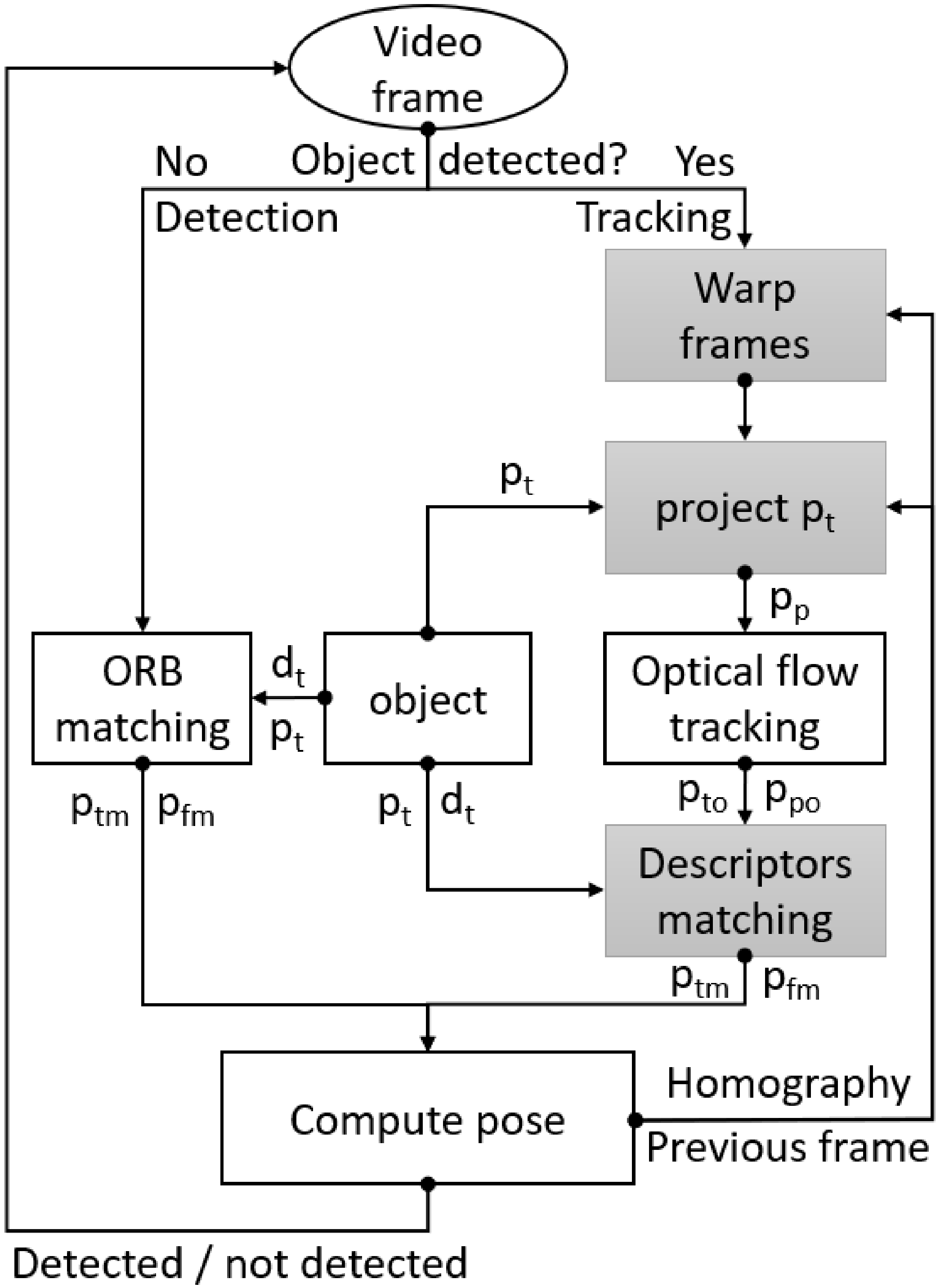
Tracking Phase

5.Compute the object pose using matched points (p_{tm} and p_{fm});

Output of each frame:

- homography
- frame image
- detection result

National Taipei University



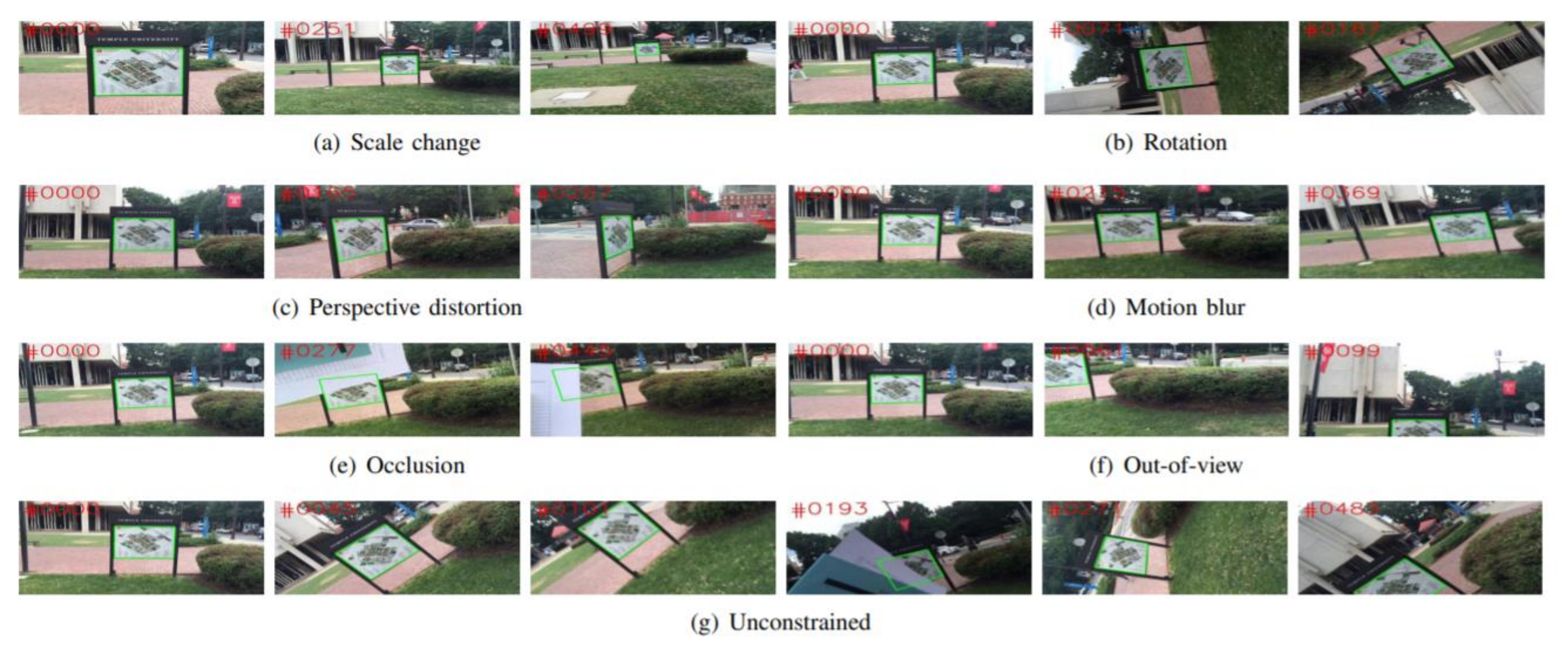
5

OBD Design Properties

- No drifting problem
- No OF points number degradation
- smoothed object pose transition between frames

OBD Evaluation

We used POT benchmark: “Planar object tracking in the wild: A benchmark” [4]



(a) Scale change (b) Rotation
(c) Perspective distortion (d) Motion blur
(e) Occlusion (f) Out-of-view
(g) Unconstrained

Benchmark

- 210 video sequences, 30 planar objects
- Every object has separated video sequences of the following scenarios:
  - scale change
  - rotation
  - perspective distortion
  - motion blur
  - occlusion
  - out-of-view
  - unconstrained

IMSLAB

Evaluation Metric: Alignment Error

- The alignment error is based on the four reference points (object corners)
- the square root of the detected points and their reference ground truth

$$e_{al} = \frac{1}{4}\sum_{i=1}^{4}\sqrt{(x_i - x_i^*)^2 + (y_i - y_i^*)^2} \qquad (5)$$

(x_i, y_i) the position of a reference corner point

(x_i^*, y_i^*) its ground truth position on the current frame.

IMSLAB

Evaluation Metric: Spatial Jitter

- We evaluate average spatial jitter as follows:

$$J_t = \sqrt{(x_i - x_{iprev})^2 + (y_i - y_{iprev})^2} \tag{6}$$

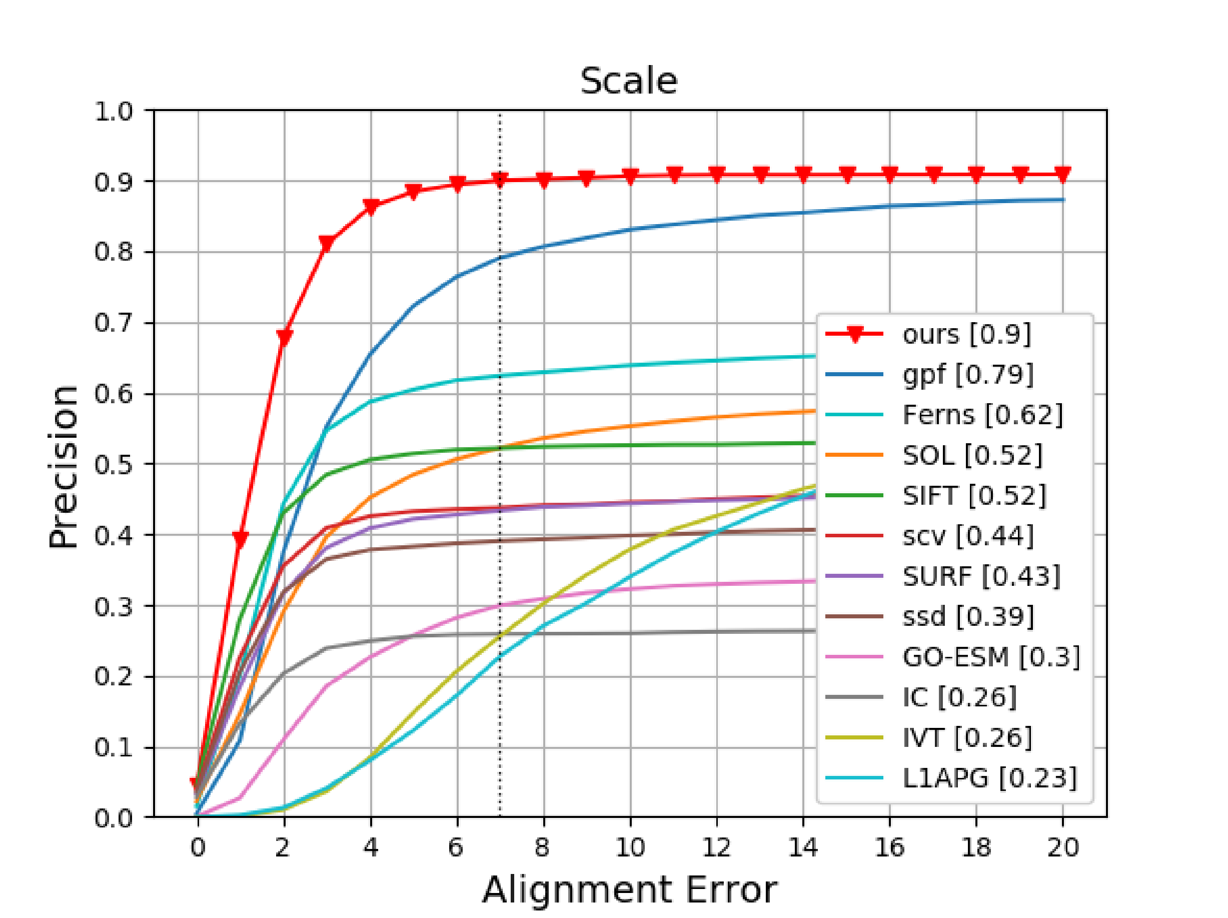
$$J^* = \sqrt{(x_i^* - x_{iprev}^*)^2 + (y_i^* - y_{iprev}^*)^2} \tag{7}$$

$$J = \frac{1}{N}\sum_{i=1}^{N} J_t - J^*. \tag{8}$$

- (x_{iprev}, y_{iprev}) the position of a reference corner point in the previous frame
- $(x_{iprev}^*, y_{iprev}^*)$ ground truth position of the corner point in the previous frame.
- J_t is tracker spatial jitter of a frame
- J^* is ground truth spatial jitter of a frame.

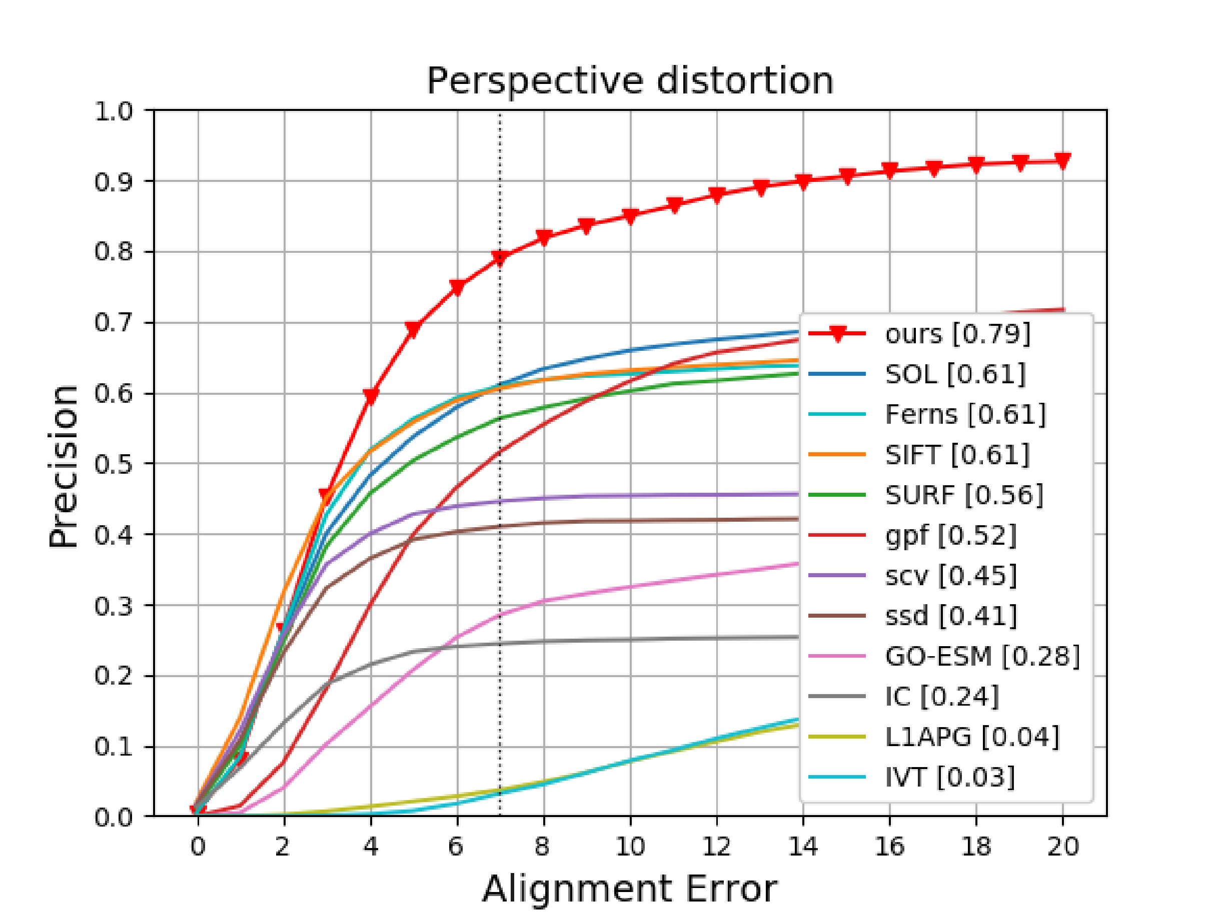
Evaluation: Scale Sequence

- Plot shows the percentage of frames (vertical axis) whose alignment error e_{AL} is smaller than the e_{AL} value on the horizontal axis
- As the representative score we use the alignment error with the threshold $t_p = 7$
- Algorithms are sorted based on the representative score in descending order



Evaluation:
Perspective Distortion Sequence

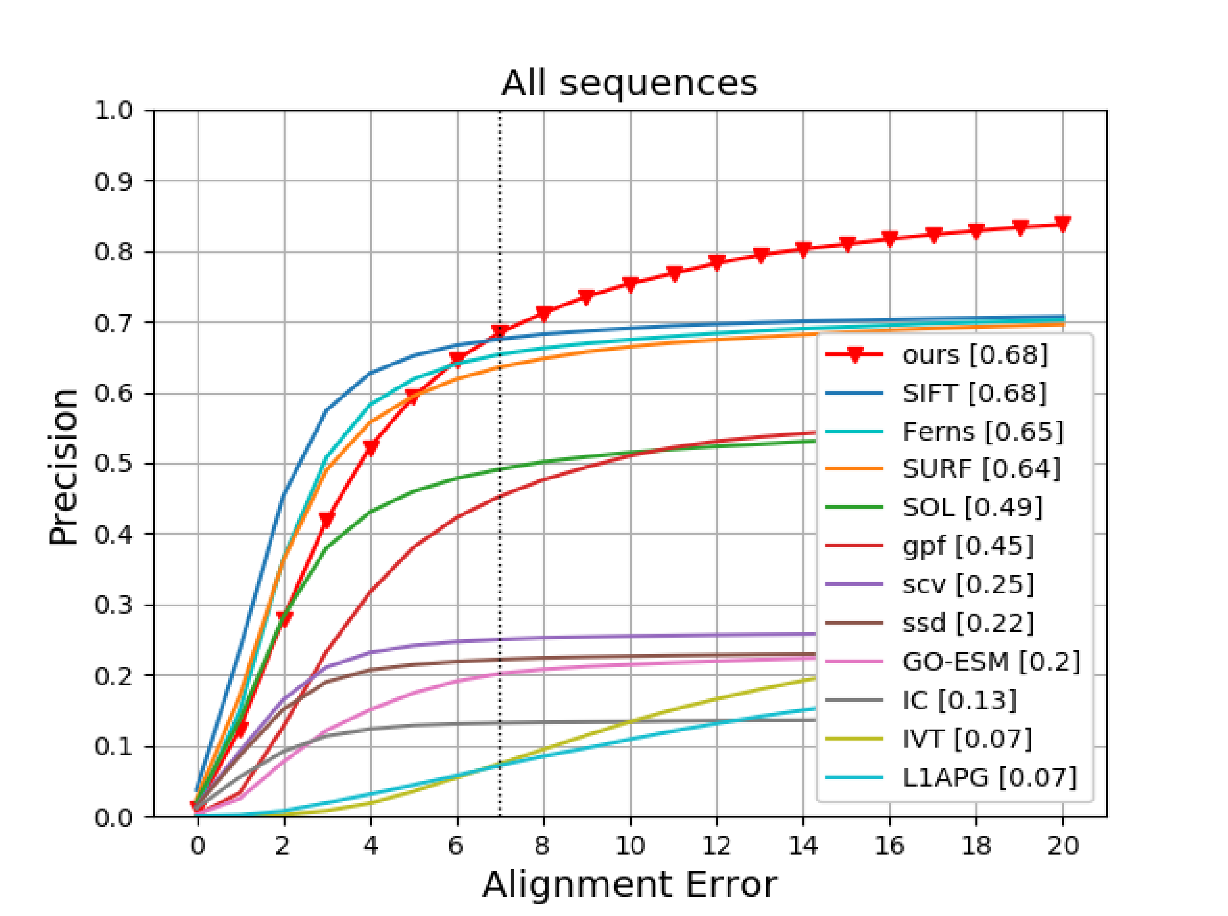
- Plot shows the percentage of frames (vertical axis) whose alignment error e_{AL} is smaller than the e_{AL} value on the horizontal axis
- As the representative score we use the alignment error with the threshold $t_p = 7$
- Algorithms are sorted based on the representative score in descending order



國立臺北大學
National Taipei University

Evaluation: All Sequences

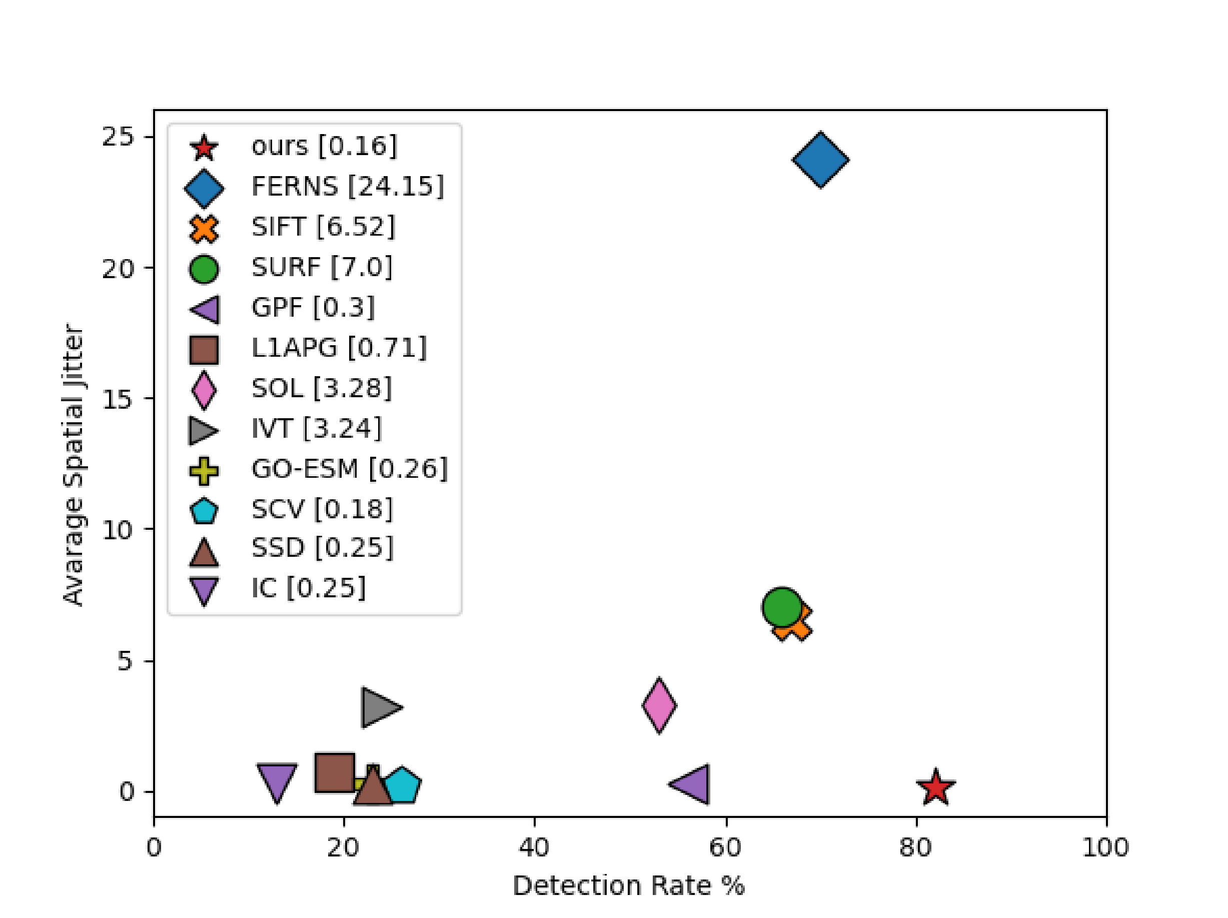
- Plot shows the percentage of frames (vertical axis) whose alignment error e_{AL} is smaller than the e_{AL} value on the horizontal axis
- As the representative score we use the alignment error with the threshold $t_p = 7$
- Algorithms are sorted based on the representative score in descending order



國立臺北大學 National Taipei University

Evaluation: Spatial Jitter

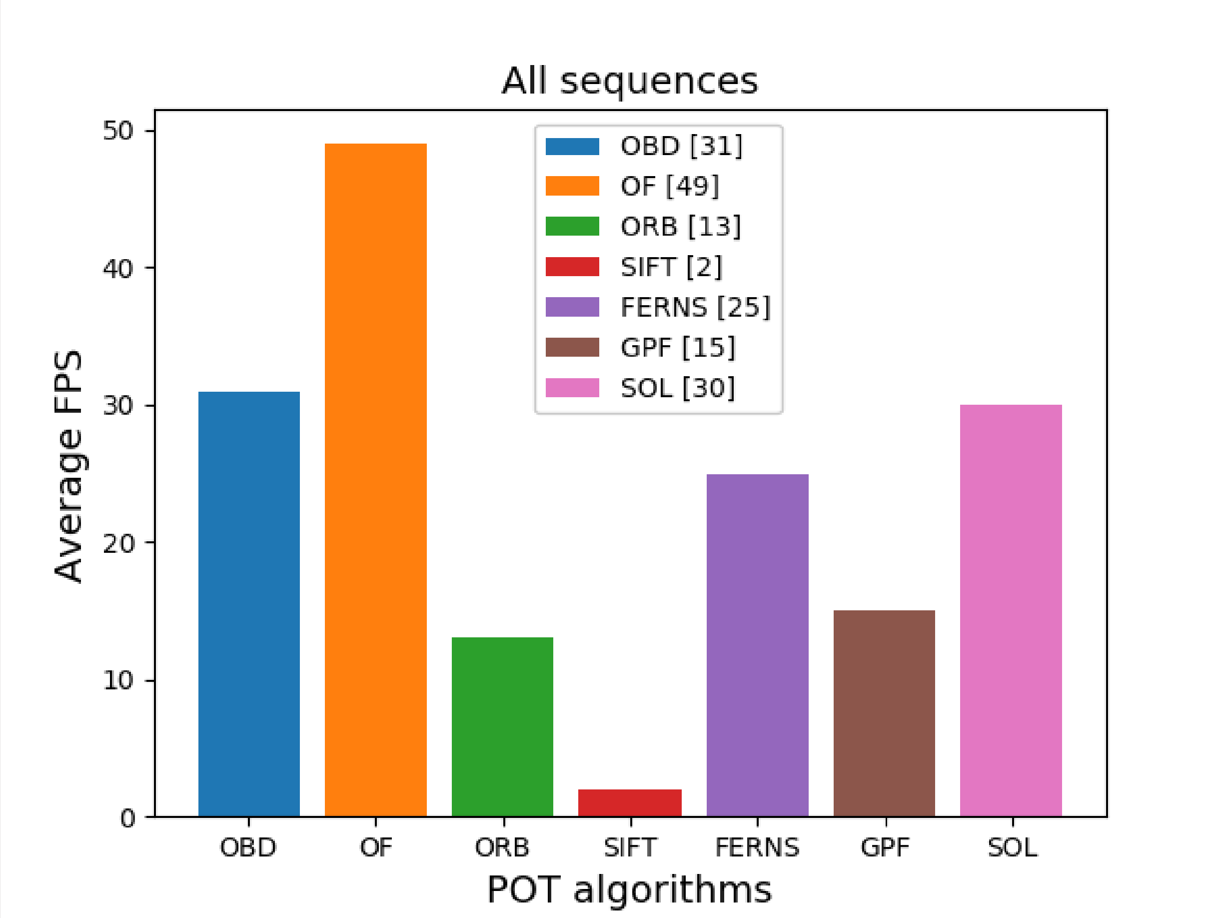
- **Detection rate:** percentage of frames with e_{AL} smaller than 20
- OBD has the highest **detection rate**
- OBD has the lowest spatial jitter



National Taipei University

Evaluation: Processing Time

- OBD achieves 30FPS on PC CPU
- With multithreading OBD achieves 30FPS on mobile phones for camera resolution 720p



國立臺北大學
National Taipei University

Evaluation: OBD vs Vuforia

- comparisons between our algorithm and Vuforia AR SDK are in supplementary materials.
- at least for small target objects Vuforia's algorithm has spatial jitter and OBD does not have it at all.




Conclusion

- OBD successfully solves the problems addressed in this study
- offers state-of-the-art precision
- OBD provides real-time mobile AR with no spatial jitter

國立臺北大學 National Taipei University

IMSLAB

Thank you for your attention!