MATLAB EXPO 2017

Development of Real-Time Object Tracking algorithm for UAVS

DEVSHREE KUMAR,(SCIENTIST,MAV UNIT,NAL,BENGULURU)

SUVARNA AGARWAL (M.TECH STUDENT)

PROBLEM STATEMENT FOR OBJECT TRACKING

- Object tracking is one of the most sought out problems in computer vision these days.

- The algorithm needs to be robust enough so that it is not restricted to a particular object.

- For object tracking basically four type of algorithms are used. Either individually or in combination with each other.
  - Feature based tracking
  - Template matching (Pattern matching)
  - Color detection and tracking (Histogram matching)
  - Edge based detection

SPEEDED-UP ROBUST FEATURES (SURF) ALGORITHM

- In this project, feature based tracking is taken in to account and SURF (Speeded-Up Robust Features) algorithm is used.

- SURF is a feature matching algorithm and MATLAB provides various commands to use this algorithm.

- The three main steps involved in the algorithm are as follows and also the commands provided by MATLAB to perform the function:
  - Detection (`detectSURFFeatures`)
  - Description (`extractFeatures`)
  - Matching (`matchFeatures`)

- SURF is scale invariant, rotation invariant and also translation invariant.

SURF ALGORITHM CONTD….

- **For detection**,

SURF uses integral form of the image and Hessian matrix as detector.

$$I_{\in}(\mathrm{x}) = \sum_{i=0}^{i\leq x}\sum_{j=0}^{j\leq y} I(i,j)$$

$$H(x,\sigma) = \begin{bmatrix} L_{xx}(x,\sigma) & L_{xy}(x,\sigma) \\ L_{yx}(x,\sigma) & L_{yy}(x,\sigma) \end{bmatrix}$$

- **For description**,

SURF uses Haar wavelets in both X and Y directions to assign the orientation to a detected point. Using this help in making the feature rotational invariant.

- **For matching**,

Surf uses the previously extracted descriptors and generate matching pairs in both images.

- **Affine transformation,**

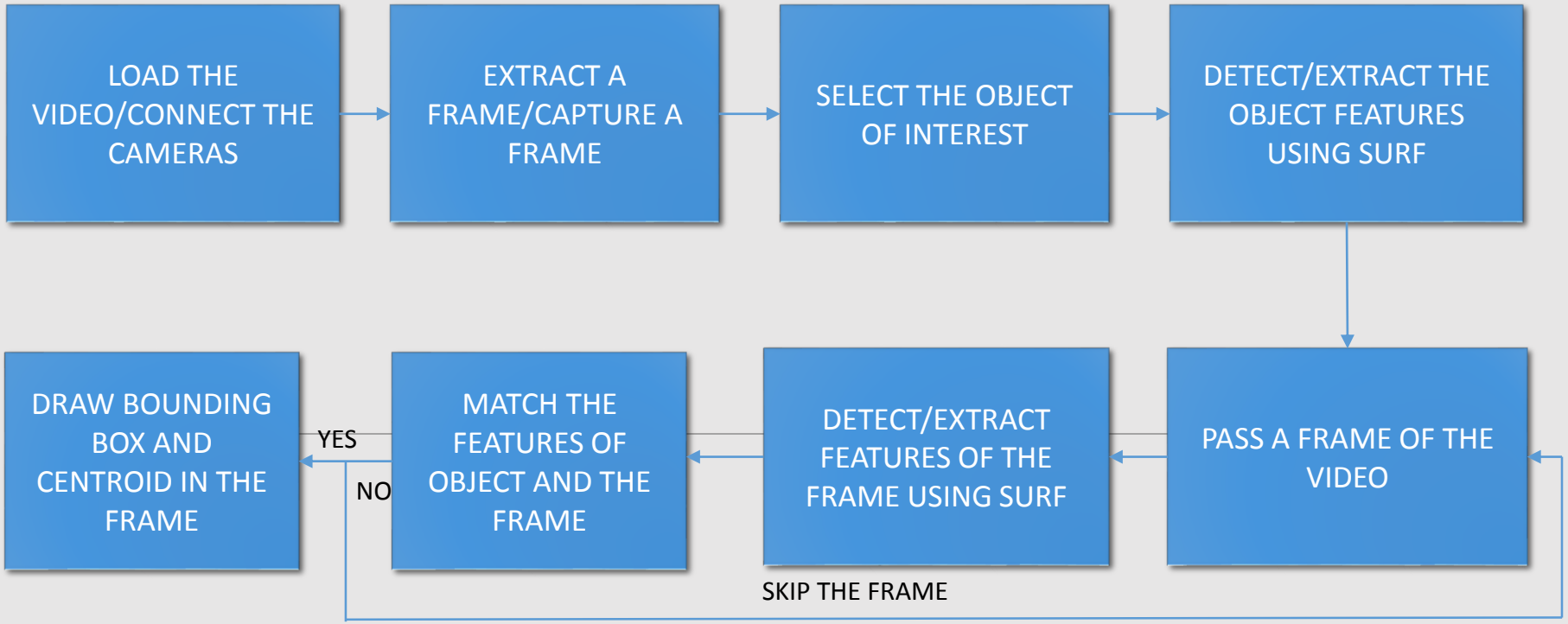
After matching, to calculate the centroid and bounding box positions, a geometrical transformation, affine, is used.

Using affine transformation the object image is transformed according to the scene image and hence centroid and bounding box positions are calculated.

$$\begin{vmatrix} x2 \\ y2 \end{vmatrix} = A \times \begin{vmatrix} x1 \\ y1 \end{vmatrix} + B$$

Here, A and B are defined according to the translation, rotational and scalar effects.

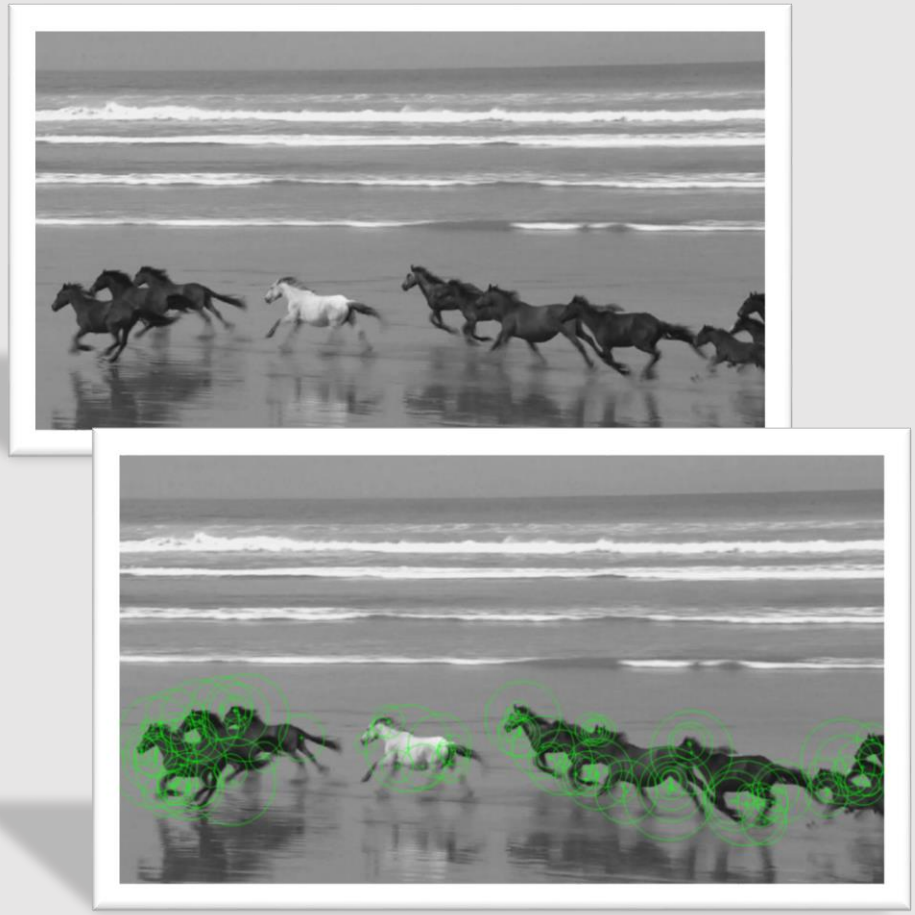
PROCESS FOR OBJECT TRACKING IN VIDEOS USING SURF ALGORITHM



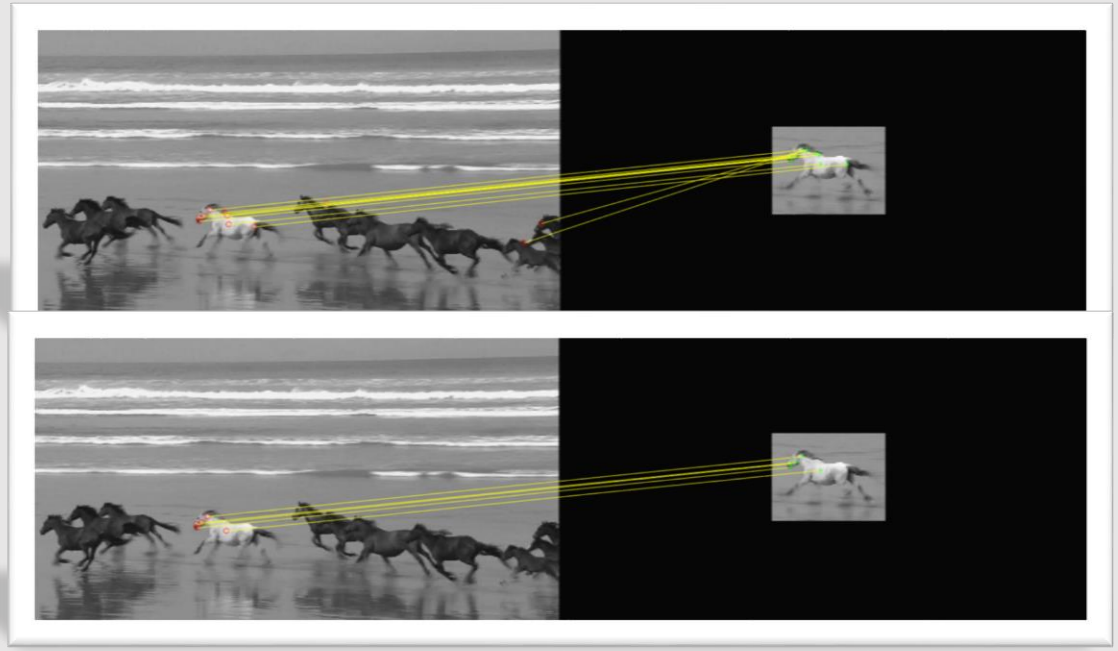
FEATURE DETECTION, DESCRIPTION AND MATCHING BY SURF

- Firstly, the object to be tracked needs to be selected.
- Here, user can select the object from the video stream itself.
- For further processing the image format is converted from RGB to grayscale.
- Using Computer vision system toolbox in MATLAB, the SURF features in the object image are detected and descriptors are assigned.

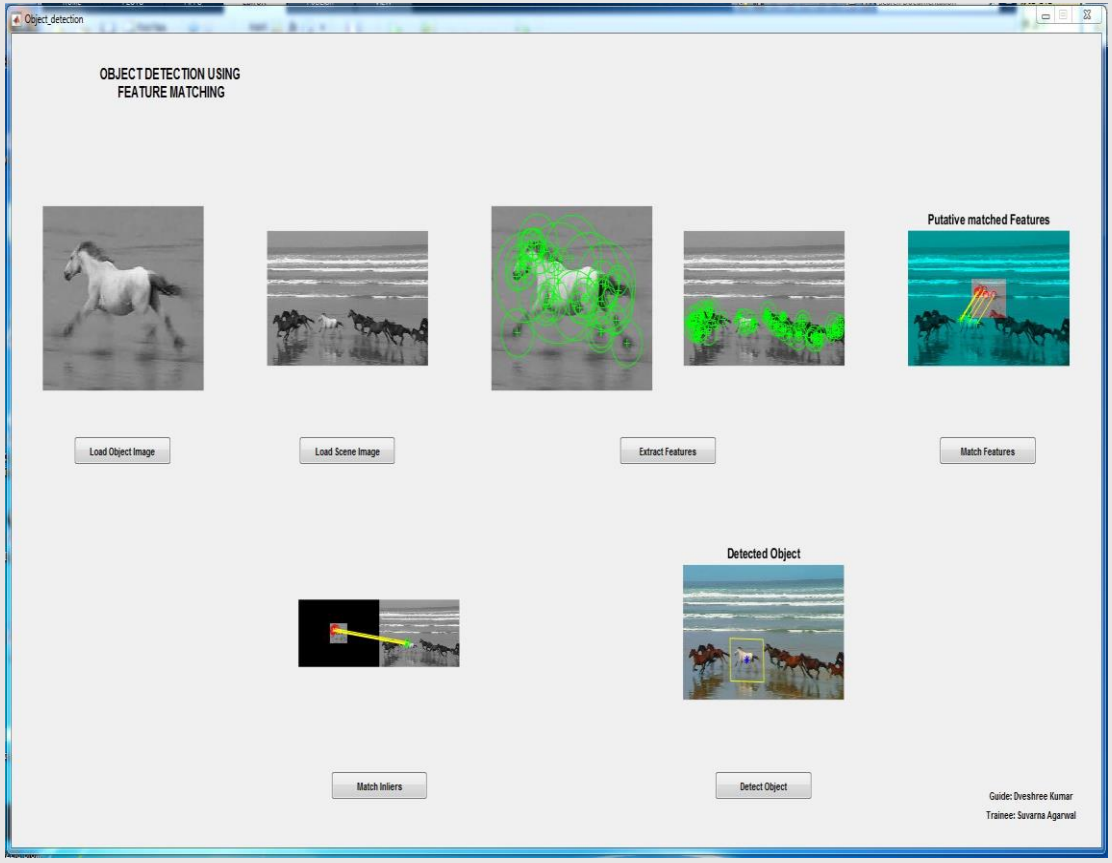
- For the next step the video stream is converted into frame sequence.
- Again, for processing the RGB format is converted into grayscale.
- Using same computer vision system toolbox in MATLAB, the SURF features in each frame is detected and descriptors are assigned.
- The number of features vary with each frame and not getting enough features can lead to certain errors.

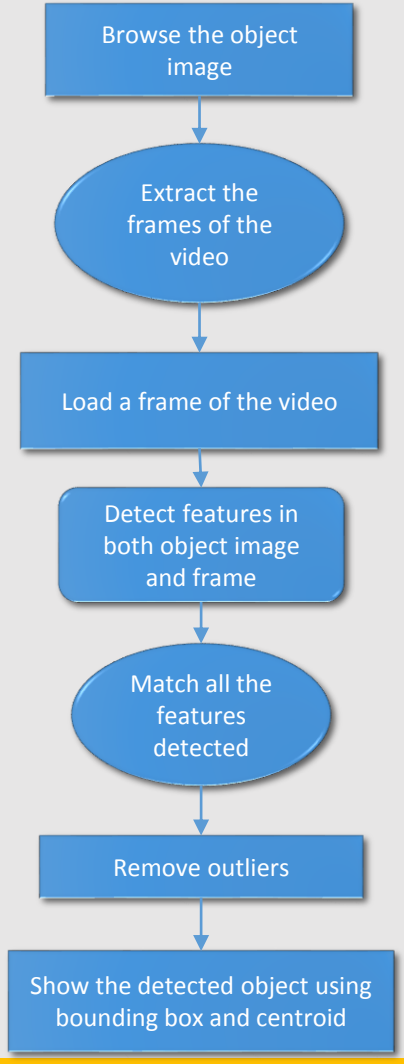


- Finally, the matching is done between the object features and each frame.
- The matching is done in two stages. First, matching of all the features then removal of outliers.
- For the frames not having enough features, are skipped and next frame is taken for processing.
- The location of the bounding box and centroid is then calculated and plotted on the video frame.

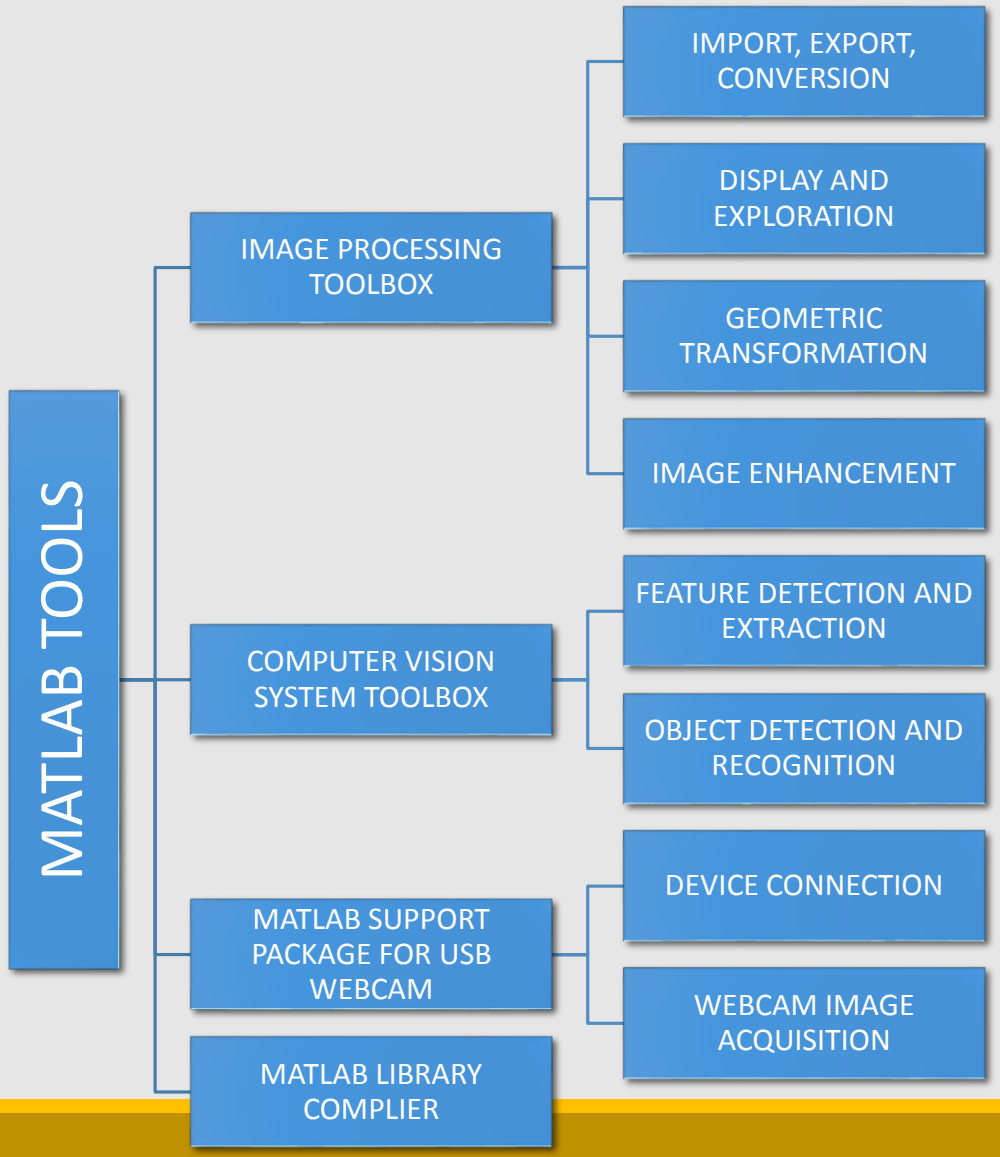


GUI IMPLEMENTATION OF SURF ALGORITHM (MATLAB)





MATLAB TOOL REQUIREMENTS



CASE:1 STATIC AND NON-DEFORMATIVE OBEJCT DETECTION AND TRACKING (NON-REALTIME VIDEO)

SELECTED OBJECT

VIDEO FRAMES

DETECTED OBJECT IN VARIOUS FRAMES

Bounding box from SURF algorithm

Bounding box from other algorithm

CASE:2 MOVING AND NON-DEFORMATIVE OBEJCT DETECTION AND TRACKING (NON-REALTIME VIDEO)

SELECTED OBJECT

VIDEO FRAMES

DETECTED OBJECT IN VARIOUS FRAMES

CASE:3 MOVING AND DEFORMATIVE OBEJCT DETECTION AND TRACKING (NON-REALTIME VIDEO)

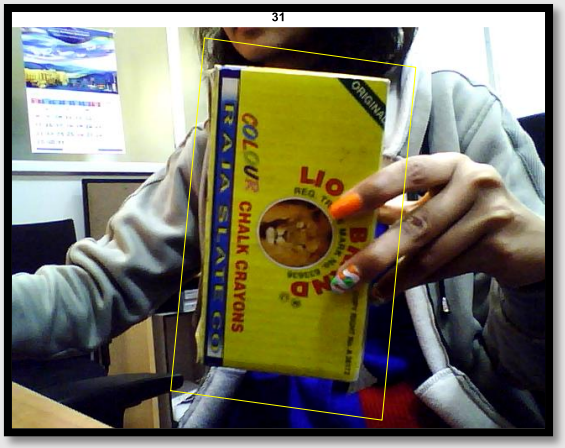
SELECTED OBJECT

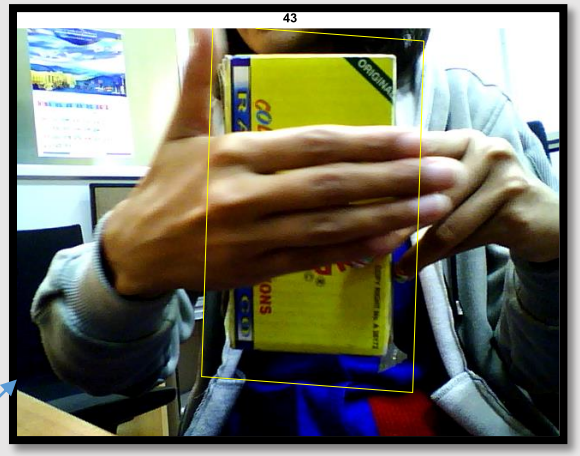
VIDEO FRAMES

DETECTED OBJECT IN VARIOUS FRAMES

CASE:4 REAL-TIME OBJECT DETECTION AND TRACKING

SELECTED OBJECT



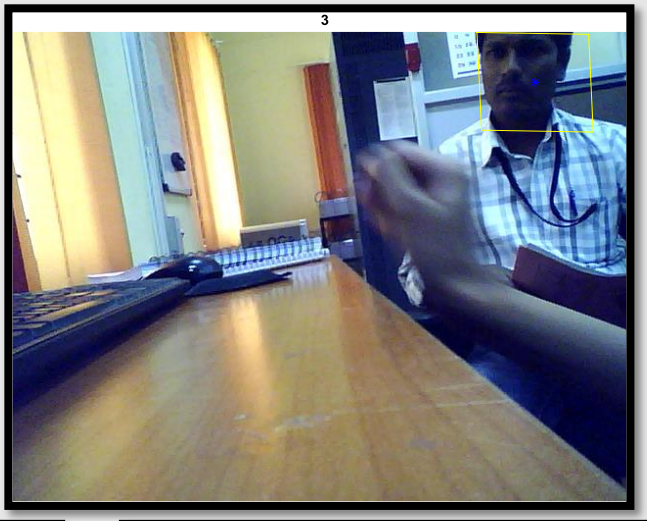


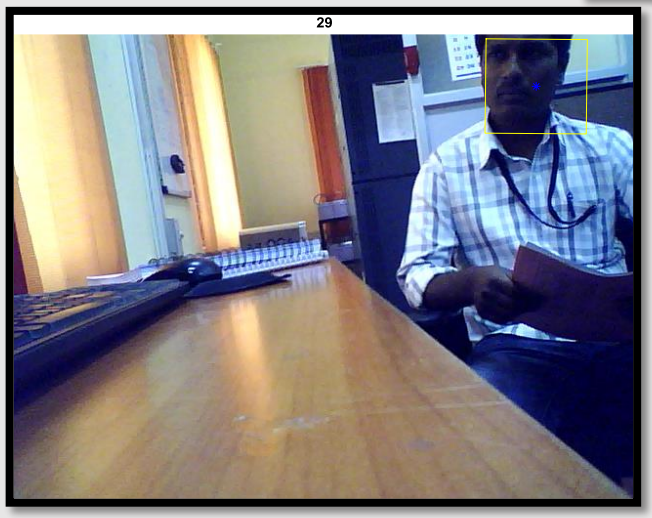
Partial Occlusion handled while performing object tracking

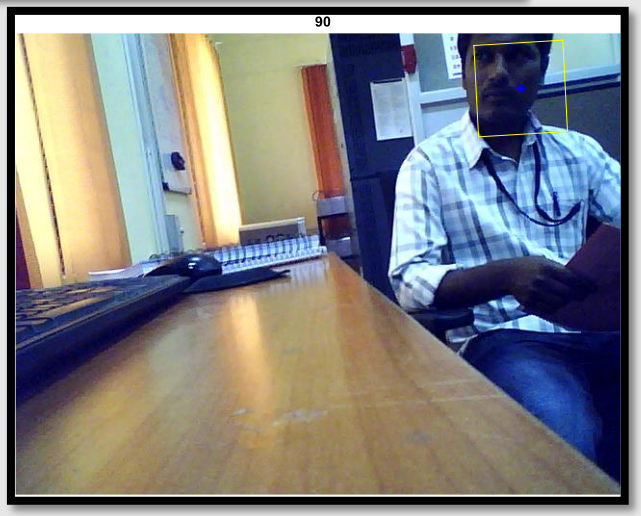
DETECTED OBJECT IN VARIOUS FRAMES

CASE:5 REAL-TIME FACE DETECTION AND TRACKING

SELECTED FACE





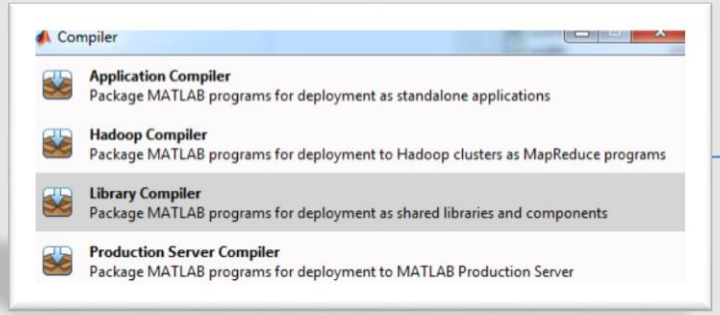


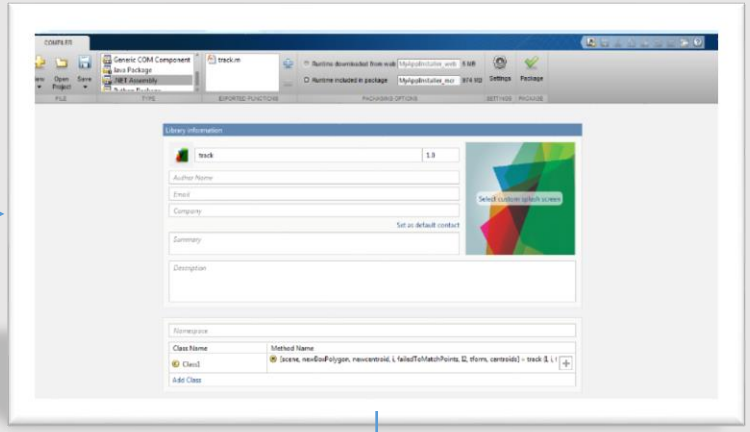
DETECTED FACE IN VARIOUS FRAMES

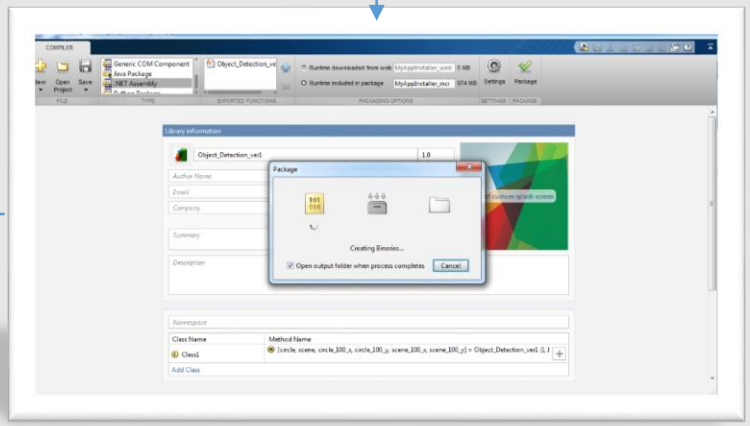
RUN-TIME OF THE TRACKING FUNCTION USED IN VARIOUS VIDEOS FOR OBJECT TRACKING

CASE	NUMBER OF FRAMES	NUMBER OF MATCHED FRAMES	OUTLIER FEATURES OF SCENE	OUTLIER FEATURES OF OBJECT	INLIER MATCHED POINTS	FUNCTION TIME FOR SINGLE FRAME (S)	TOTAL FUNCTION TIME (S)
1	100	41	22345	540	272	0.092	3.464
2	100	100	49152	8000	1905	0.099	6.688
3	100	85	39241	3312	613	0.096	10.693
4	100	79	59109	28576	2269	0.136	12.608
5	100	70	31856	4680	717	0.113	6.511

CONVERSION OF USER-DEFINED MATLAB FUNCTION IN .DLL USING MATLAB COMPILER



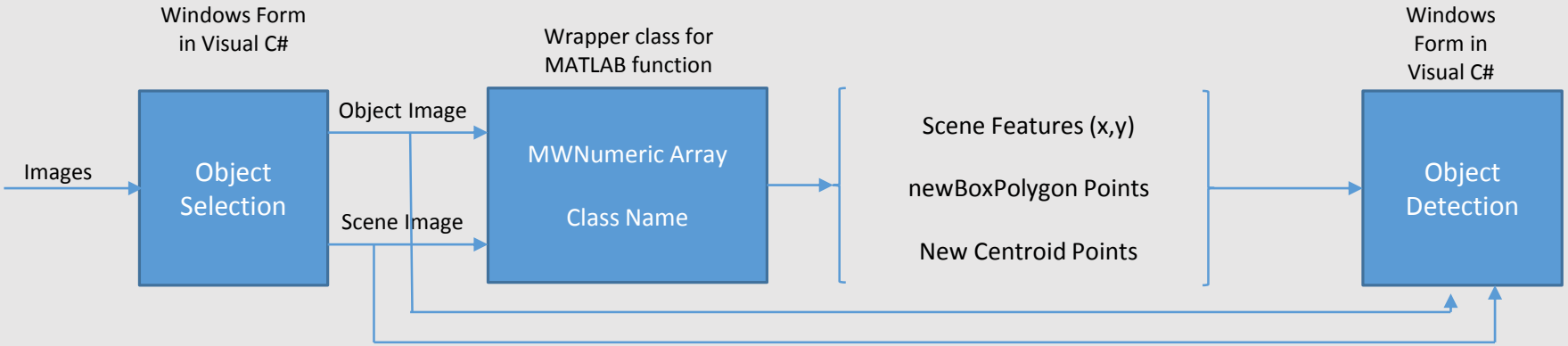




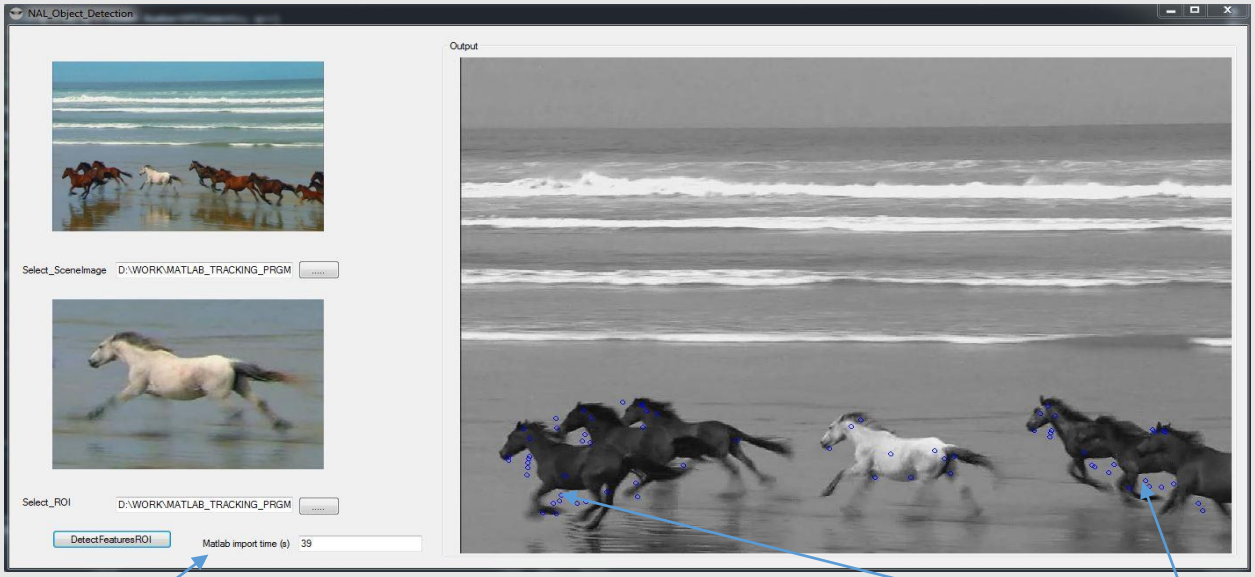
2 .dlls are created which are then added to Visual C# as follows

- Adding References in Visual C#
  - MWArray.dll
  - *.dll
- Adding MATLAB libraries in Visual C#
  - MathWorks.MATLAB.NET.Array
  - MathWorks.MATLAB.NET.Utility
  - MATLAB created function

MATLAB .dll IMPLEMENTAION IN VISUAL C#



Work-flow of MATLAB implementation in Visual C#



Object detection using SURF in Visual C#

ONGOING WORK