FLORIDA ATLANTIC UNIVERSITY

DATA SCIENCE, ANALYTICS, AND ARTIFICIAL INTELLIGENCE CONFERENCE

SATURDAY
NOVEMBER 14, 2020

FAU.EDU/DATA

BREAKOUT SESSION

PRESENTATION DESCRIPTION

CHARLES E. SCHMIDT COLLEGE OF SCIENCE

fau.edu/data

Smart Track Project Review (DRAFT)

Anatoly Vasiliev Ph.D.

Consultant, DMG LLC

August 10, 2020

Project Outline

1. Application Scenarios and Their Implementation
   - Review and analyze requirements for DMG's PDT demo application for tracking people in a 3D scene
   - Measure distances between the people
   - detect undesirable contacts (distance between people gets shorter than 6 ft)
   - Measure the duration of the undesirable contacts
2. People Detection and Tacking
   - Select among existing MATLAB functions and applications and modify/tailor them to meet the DMG Smart Track Application scenarios
3. People Recognition/ID
   - Use computer vision to extract discriminative information from facial images
   - Use pattern recognition or machine learning techniques to model the appearance of faces and to classify them.

Application Scenarios: Contact Tracking with 1 Static and 2 Moving Objects

Description: Identity instances when any of 2 moving objects (humans) happens to get less than 6 feet (2 meters) apart from the static object for at least 2 consecutive minutes.

Primary Actor: Statically positioned reference object labeled Person 1 (Marker).

Preconditions: A camera captures area within at least 3m radius behind the statically placed object. Surveillance duration time is 1 hour. The static object keeps his/her position unchanged during the surveillance period.

Postconditions: Report start time, duration and the contact IDs for each instance.

Application Scenarios: Contact Tracking with 1 Static and 2 Moving Objects (continued 1)

1. **'Happy Path' Scenario**: 9:00 am – Person 1 takes a seat in front of the computer with a camera, capturing area around (min radius 3m)
   - 9:02 am – another person (Person 2) comes and stays at a distance <2m from Person#1 for 2 min;
   - 9:04 am – Person 2 leaves the surveillance area;
   - **Expected result**: Reporting one contact instance between Person #1 and Person#2, contact start time 9:02 am, contact duration 2 min.
2. **Scenario 1a**: 9:00 am – Person 1 takes a seat in front of the computer with a camera, capturing area around (min radius of observation is 3m)
   - 9:02 am – Person 2 comes in and moves for 2 min around Person 1 but does not come closer that at 2m from Person 1;
   - 9:04 am – Person 2 comes closer and stays next to Person#1 for the next 3 min
   - 9:07 am – Person #2 leaves surveillance area;
   - **Expected result**: Reporting one contact instance between Object 1 and 2, contact start time 9:04 am, contact duration 3 min.

Application Scenarios: Contact Tracking with 1 Static and 2 Moving Objects (continued 2)

3. **Scenario 1b**: 9:00 am – Person #1 takes a seat in front of the computer with camera, capturing area around (min radius of observation is 3m)
   3. 9:02 am – Person 2 comes and moves around Person 1 for 2 min keeping at a distance greater than 2m from Person 1;
   4. 9:04 am – Persoon#2 leaves surveillance are;
   5. Expected result: Reporting zero contact instances.
4. **Scenario 1c**: 9:00 am – Person 1 takes a seat in front of the computer with camera, capturing area behind Person 1 (min radius 3m)
   - 9:02 am – Person 2 comes and moves for 2 min around Person 1 keeping at a distance greater than 2m from Person 1;
   - 9:04 am – Person 2 comes closer (at a distance less than 2m) and stays next to Person#1 for the next 3 min;
   - 9:07 am – Person 2 moves away and stays 2.5 m apart from Person#1 for 1 min;
   - 9:08 am – Person 2 comes back, moves within 1-1.5 m from Person 1, and stays there for 3 min;
   - 9:11am Person 2 leaves the surveillance area;
   - **Expected result**: Reporting two contact instances between Persons 1 and 2;
   - **Instance #1**: contact start time 9:04 am, contact duration 3 min;
   - **Instance#2**: contact start time 9:08 am, contact duration 3 min.

Application Scenarios: Contact Tracking with 1 Static and 2 Moving Objects (continued 3)

5. **Scenario 1d**: 9:00 am - Person 1 takes a seat in front of the computer with a camera, capturing area around (min radius observation is 3m).
    - 9:02 am – Person 2 comes and moves for 1 min around Person #1 but stays > 2m apart from Person 1
    - 9:04 am – Person 2 comes closer (at a distance <2m) and stays next to Person 1 for the next 1 min
    - 9:07 am – Person 2 moves away and stays 2.5 m apart from Person 1 for 1 min
    - 9:08 am – Person2 comes back and moves within 1-1.5 m from Person 1 for 3 min
    - 9:11 am Person 2 leaves the surveillance area
    - Expected result: Reporting one contact instance between Object 1 and 2: contact start time 9:08 am, contact duration 3 min

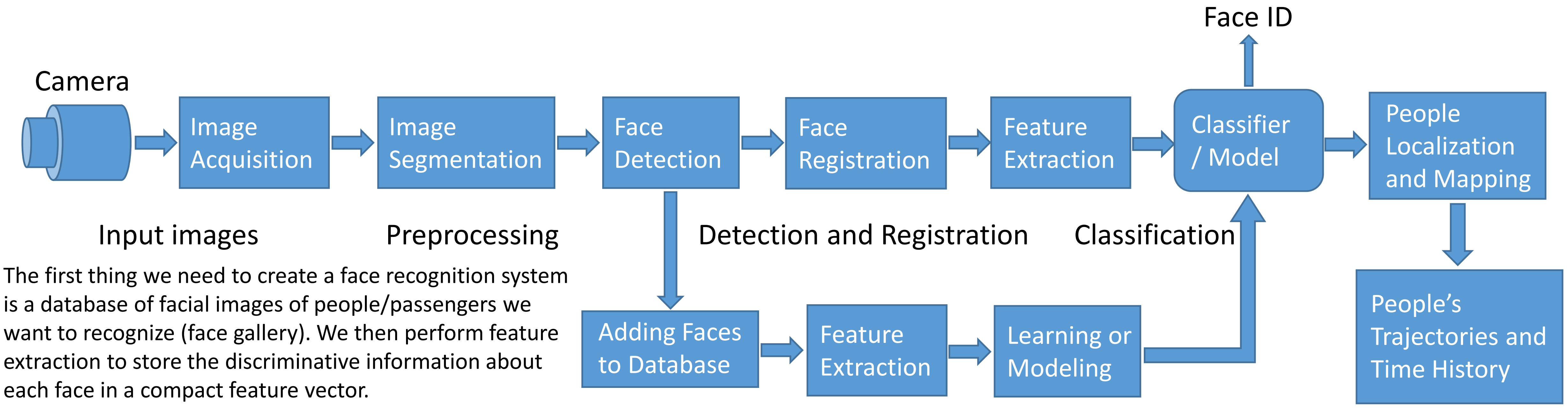
Application Scenarios: Contact Tracking with 1 Static and 2 Moving Objects (continued 4)

6. **Scenario 2a**: 9:00 am – Person #1 takes a seat in front of the computer with a camera, capturing area behind (min radius of observation is 3m).
   - 9:02 am – Person 2 comes and moves for 2 min around Person 1 but does not come closer that 2m apart from Person1.
   - 9:04 am – Person 2 comes closer and stays at less than 2m from Person#1 for the next 3 min
   - 9:07 am – Person 2 leaves the surveillance area
   - 9:08 am – Person 3 comes closer and stays next to Person 1 for the next 3 min
   - 9:11am Person 3 leaves the surveillance area.
   - **Expected result**: Reporting 2 contact instances:
   - **Instance 1**: a contact between Person 1 and Person 2, which starts at 9:02am and lasts for 3 min
   - **Instance 2**: a contact between Person 1 and Person 3, which starts at 9:08 am and lasts 3 min
7. **Scenario 3a**: 9:00 am – Person 1 takes a seat in front of the computer with a camera, capturing area around (min radius of observation is 3m).
   - 9:02 am – Person 2 comes and stays closer than at 2m from Person#1 for the next 3 min
   - 9:03 am – Person 3 comes and stays closer than at 2m from Person#1 for the next 3 min
   - 9:05 am Person 2 leaves the surveillance area
   - 9:06 am Person 3 leaves the surveillance area
   - **Expected result**: Reporting 2 contact instances:
   - **Instance 1**: a contact between Person 1 and Person 2, which starts at 9:02amand lasts 3 min
   - **Instance 2**: a contact 2 between Person 1 and Person 3, which starts at 9:03 am and lasts 3 min

People Detection and Tracking

PDT algorithm implemented in MATLAB

PDT Algorithm Layout and Workflow



The first thing we need to create a face recognition system is a database of facial images of people/passengers we want to recognize (face gallery). We then perform feature extraction to store the discriminative information about each face in a compact feature vector.

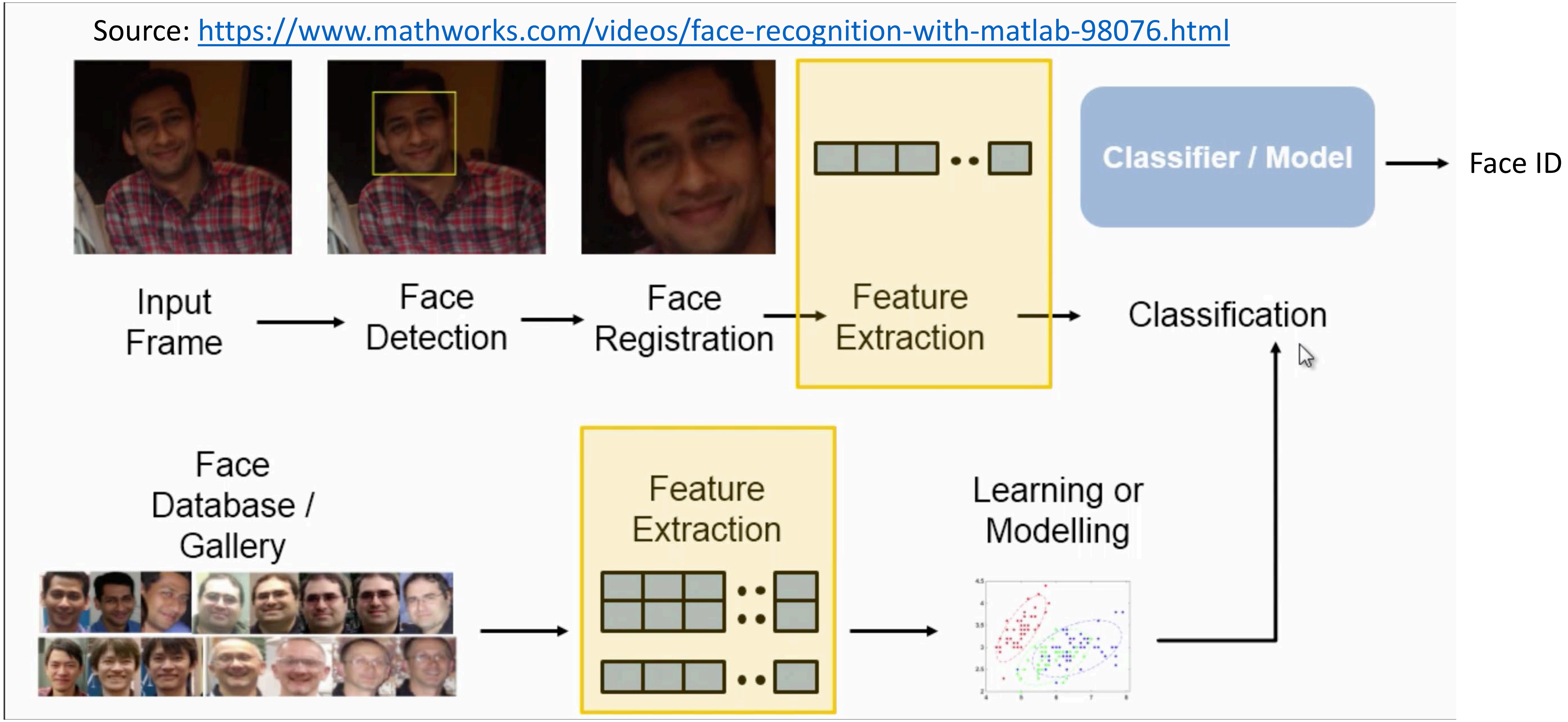
Following the feature extraction, we have a learning or modeling step when a machine learning algorithm is used to fit a model of the appearance of the faces in the gallery, so that we can discriminate between faces of different people in the database. The output of this stage is a classifier, a model that will be used to recognize and label input images.

With an input query image present, a face detection algorithm is used to find where the faces are located in the image. We then crop, resize, and normalize the face to match the size and pose of images used in the training face gallery. Then form the same feature extraction is performed that was done with the face gallery, and we run that through our classifier or model. The output is a label or an indicator to signify which person from the database the query image belongs to.

Localization and mapping procedure calculates people's trajectories, time histories, and identifies the unwanted crossing events (coming at a distance sorter than recommended for a longer than allowed time).

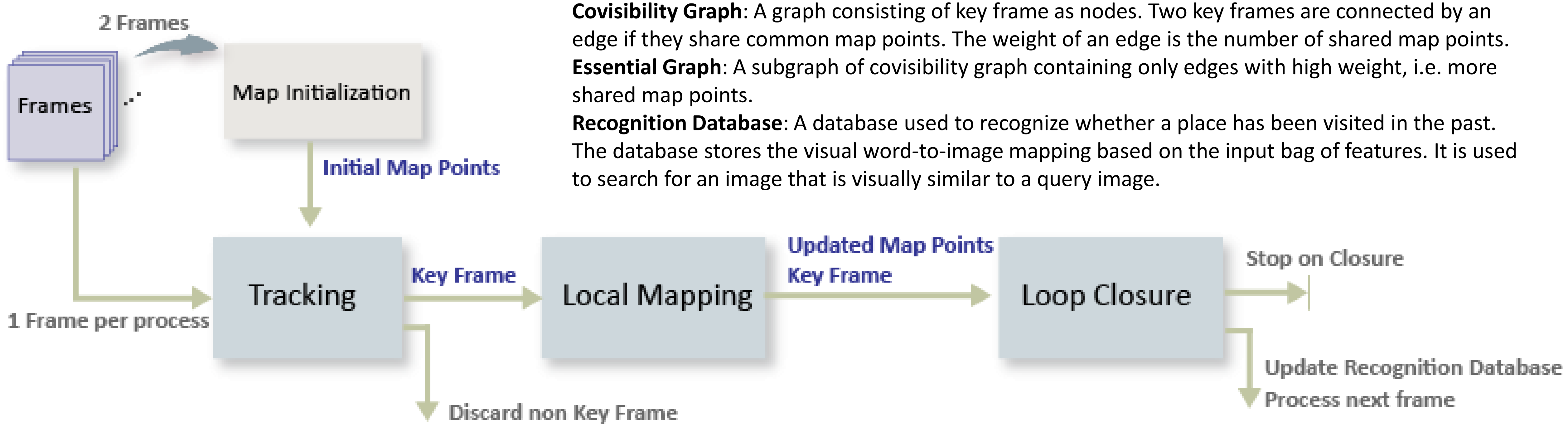
Face Detection and Identification

Source: https://www.mathworks.com/videos/face-recognition-with-matlab-98076.html



Monocular Visual Simultaneous Localization and Mapping

Key Frames: A subset of video frames that contain cues for localization and tracking. Two consecutive key frames usually involve sufficient visual change.
Map Points: A list of 3-D points that represent the map of the environment reconstructed from the key frames.
Covisibility Graph: A graph consisting of key frame as nodes. Two key frames are connected by an edge if they share common map points. The weight of an edge is the number of shared map points.
Essential Graph: A subgraph of covisibility graph containing only edges with high weight, i.e. more shared map points.
Recognition Database: A database used to recognize whether a place has been visited in the past. The database stores the visual word-to-image mapping based on the input bag of features. It is used to search for an image that is visually similar to a query image.



https://www.mathworks.com/help/vision/examples/monocular-visual-simultaneous-localization-and-mapping.html

Calculated Object Trajectory

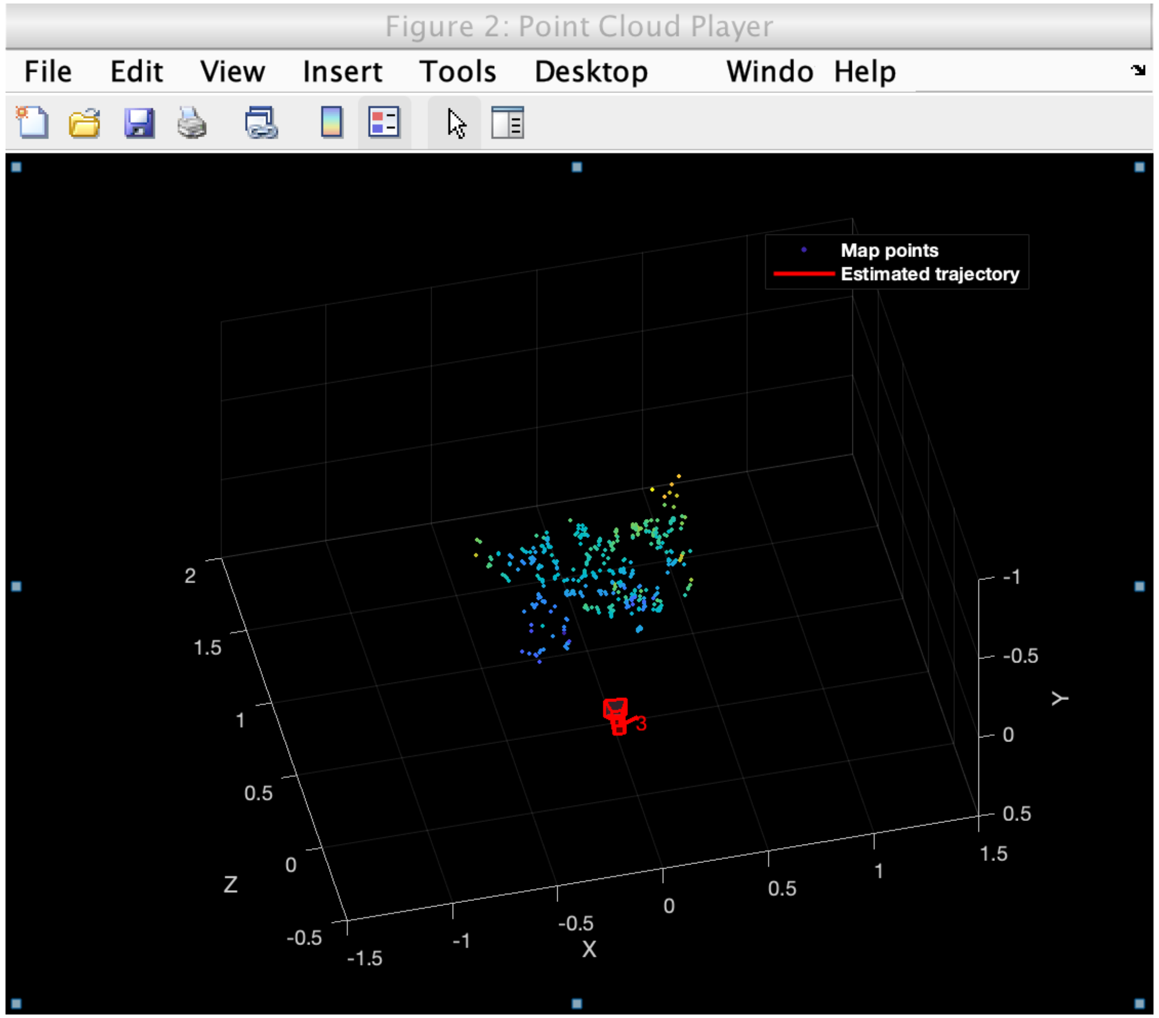
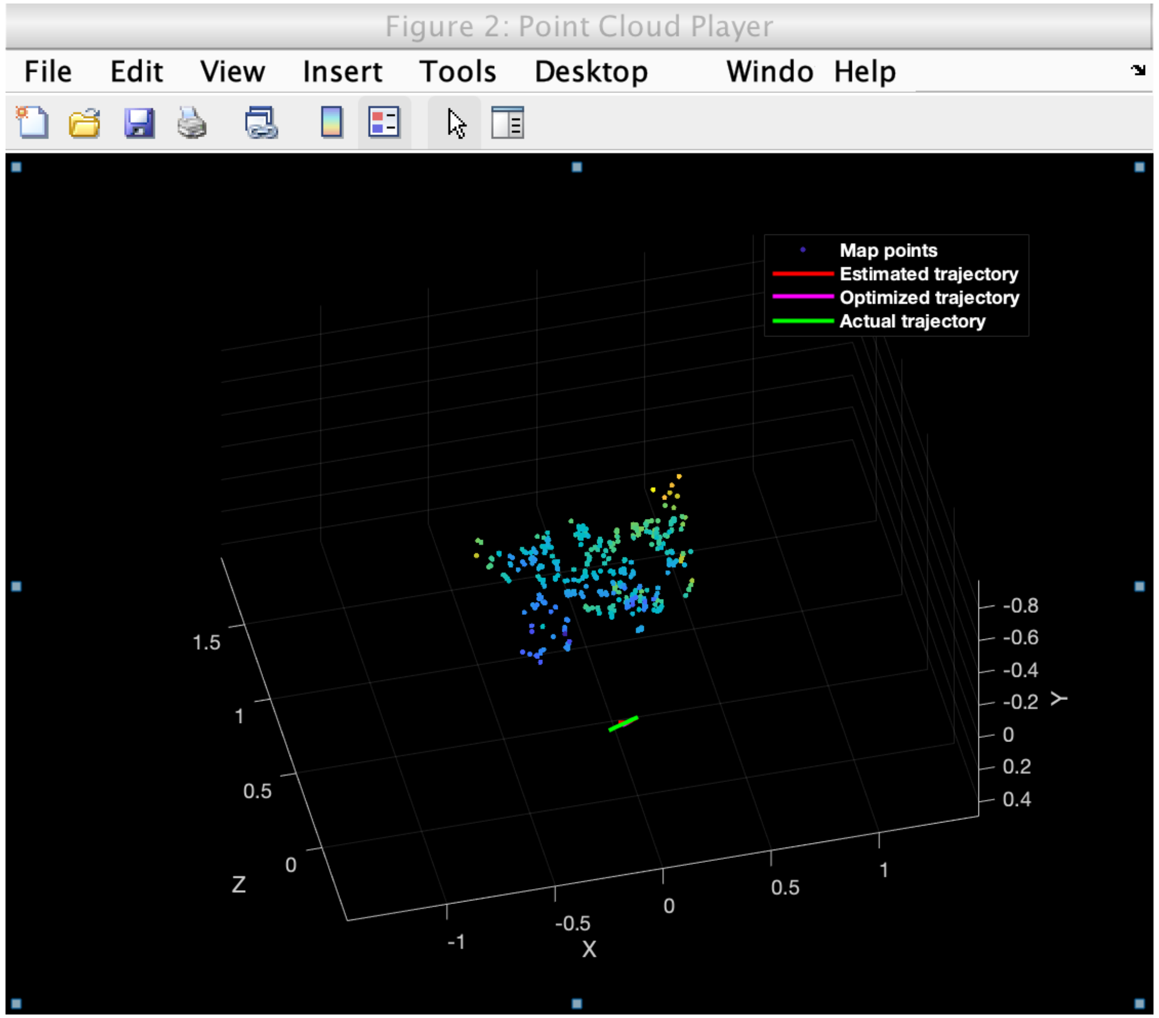


Figure 2: Point Cloud Player
File Edit View Insert Tools Desktop Windo Help
Map points
Estimated trajectory
X
Y
Z
-1.5
-1
-0.5
0
0.5
1
1.5
2



MATLAB Functions and Scripts Modified and Used with PDT

Selected among existing MATLAB functions and scripts, modified and used in the PDT demo

List of Selected MATLAB Functions and Scripts

Function/Script name	**Description**	**Toolboxes used**
FeatureBasedObjectDetectionExample.m	Object detection in clutter	Computer Vision
SimpleFaceRecognitionMathWorks.m	Face recognition using feature extraction	Statistics and Machine Learning
detectAndTrackFaces.m	Face detection and tracking	Image Acquisition, Computer Vision
MultiObjectTrackerKLT.m	Tracking multiple objects	Image Processing, Computer Vision
MonocularVisualSimultaneousLocalizationAndMappingExample.mlx	Visual simultaneous localization and mapping (vSLAM)	Statistics and Machine Learning, Deep Learning
visionfacetracking.m	Face Detection and Tracking Using CAMShift	Image Acquisition, Computer Vision

Useful links:
MathWorks Detection, Tracking and Computer Vision Toolbox Algorithms

1. Face/Nose/Eye/Mouth and Torso Detector: https://www.mathworks.com/help/vision/ref/vision.cascadeobjectdetector-system-object.html
2. Kalman Filter motion-based tracking examples
   - Motion-Based Multiple Object Tracking: https://www.mathworks.com/help/vision/examples/motion-based-multiple-object-tracking.html
   - Tracking Pedestrians from a Moving Car: https://www.mathworks.com/help/driving/examples/track-pedestrians-from-a-moving-car.html
   - Using Kalman Filter for Object Tracking: https://www.mathworks.com/help/vision/examples/using-kalman-filter-for-object-tracking.html
3. Date and Time Arithmetic capabilities with timestamps in base MATLAB, calculating elapse time: https://www.mathworks.com/help/matlab/matlab_prog/compute-elapsed-time.html
4. Semantic Segmentation Using Deep Learning: https://www.mathworks.com/help/vision/examples/semantic-segmentation-using-deep-learning.html
5. Sensor fusion for object detection, localization and tracking: https://www.mathworks.com/help/matlab/matlab_prog/compute-elapsed-time.html https://www.mathworks.com/help/fusion/examples.html?s_v1=29589&elqem=2966950_EM2_WW_NUR_20-01_SENSOR-FUSION-NURTURE-PILOT
6. Oxford University detection and tracking SORT project: https://github.com/ZidanMusk/experimenting-with-sort

Landing AI tool to monitor social/physical distancing in the workplace: https://landing.ai/landing-ai-creates-an-ai-tool-to-help-customers-monitor-social-distancing-in-the-workplace/
Andrew Ng's personal web page at Stanford University: https://www.andrewng.org/

DMGsmart

MATLAB Functions and Scripts Tested and Modified

Test and modify the existing MATLAB functions and scripts

MATLAB Functions and Scripts Tested and Modified

Function/Script name	Description	Toolboxes used	Test results
AcquireOneImageFrameFromWebcamExample.mlx	Acquiring an image frame	Image Acquisition, Computer Vision	Modified
DateAndTimeArithmeticExample.mlx	Date and time arithmetic	Basic MATLAB	Tested OK
detectAndTrackFaces.m,	Face detection and tracking	Image Acquisition, Computer Vision	Modified
DetectFacesFrontalFaceClassificationModExample.mlx	Detect faces in images	Image Processing, Computer Vision	Tested OK
FeatureBasedObjectDetectionExample.m	Object detection in clutter	Computer Vision, Statistics and Machine Learning	Tested OK
kalmanFilterForTracking.m	Tracking multiple objects	Image Processing, Computer Vision	Tested OK
MonocularVisualSimultaneousLocalizationAndMappingExample.mlx	Visual simultaneous localization and mapping (vSLAM)	Statistics and Machine Learning, Deep Learning	To be rewritten
MotionBasedMultiObjectTrackingExample.m	Tracking multiple objects	Image Acquisition, Computer Vision	Tested OK
MultiObjectTrackerKLT.m	Tracking multiple objects	Image Processing, Computer Vision	Tested OK
PedestrianTrackingFromMovingCameraExample.m	Tracking multiple objects	Image Processing, Computer Vision	Modified
SemanticSegmentationUsingDeepLearningExample.mlx	A semantic segmentation network classifier	Deep Learning, Computer Vision	To be rewritten
SimpleFaceRecognitionMathWorks.m	Face recognition using feature extraction	Statistics and Machine Learning	To be rewritten
StructureFromMotionFromMultipleViewsExample.mlx	Estimating 3-D structure using SfM procedure	Image Acquisition, Computer Vision	Tested OK
TrainAYOLOV2VehicleDetectorExample	Train a detection algorithm based on a YOLO v2 network	Statistics and Machine Learning, Deep Learning	Tested OK
visionfacetracking.m, visionfacetracking.mlx	Face Detection and Tracking Using CAMShift	Image Acquisition, Computer Vision	Tested OK

- Geometric camera calibration, also referred to as camera resectioning, is an essential part of the Image Acquisition and Computer Vision toolboxes, which estimates the parameters of a lens and image sensor of an image or video camera. PDT will use these parameters to correct for lens distortion, measure the size and distances between objects in world units, and determine the location of the objects with respect to the camera in the scene. https://www.mathworks.com/help/vision/ug/camera-calibration.html
- Automating image registration (ground truth) with MATLAB: https://www.mathworks.com/company/newsletters/articles/automating-image-registration-with-matlab.html
- Some portions of the useful MATLAB functions and scripts that did not pass the tests and could not be modified will be rewritten and used in the main PDT MATLAB code.

DMGsmart

MATLAB Code to Integrate the Modified Functions and Scrips into the PDT Application

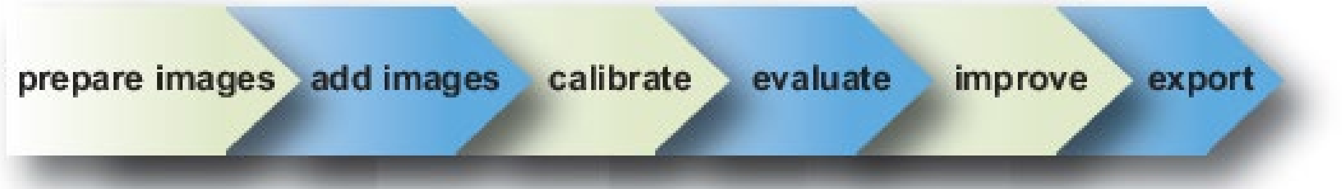
DMGSMAR-28: Write MATLAB code to integrate the modified functions and scrips into the PDT – Part 1

PDT Algorithm

1. PDT application involves video stream acquisition, recording a video, multiple object (human) detection, tracking, localization (objects' distances from the camera and and their azimuth angles), and timing.
   - **Acquisition**: acquire a video file with a calibrated camera
   - **Detection**: detect objects of interest in a video frame.
   - **Prediction**: predict the object locations in the next frame.
   - **Data association**: use the predicted locations to associate detections across frames to form *tracks*.
   - **Timing**: use Date and Time Arithmetic capabilities with timestamps, calculate elapse time.
2. Objects' trajectories with timestamps will be stored in a history file and used to calculate distances between objects, start time, duration and the contact IDs for each instance when any of two objects comes and stays at a distance <2m from each other for longer than 2 min.

Camera Calibration Procedure

The **Camera Calibrator** app estimates camera intrinsic parameters, extrinsic parameters, and lens distortions. We use these camera parameters for our Smart Track computer vision application. The camera calibration provides removing the effects of lens distortion from an image, measuring planar objects, or reconstructing 3-D scenes from multiple views captured with a mono camera.
The suite of calibration functions used by the **Camera Calibrator** app provide the following workflow for camera calibration.

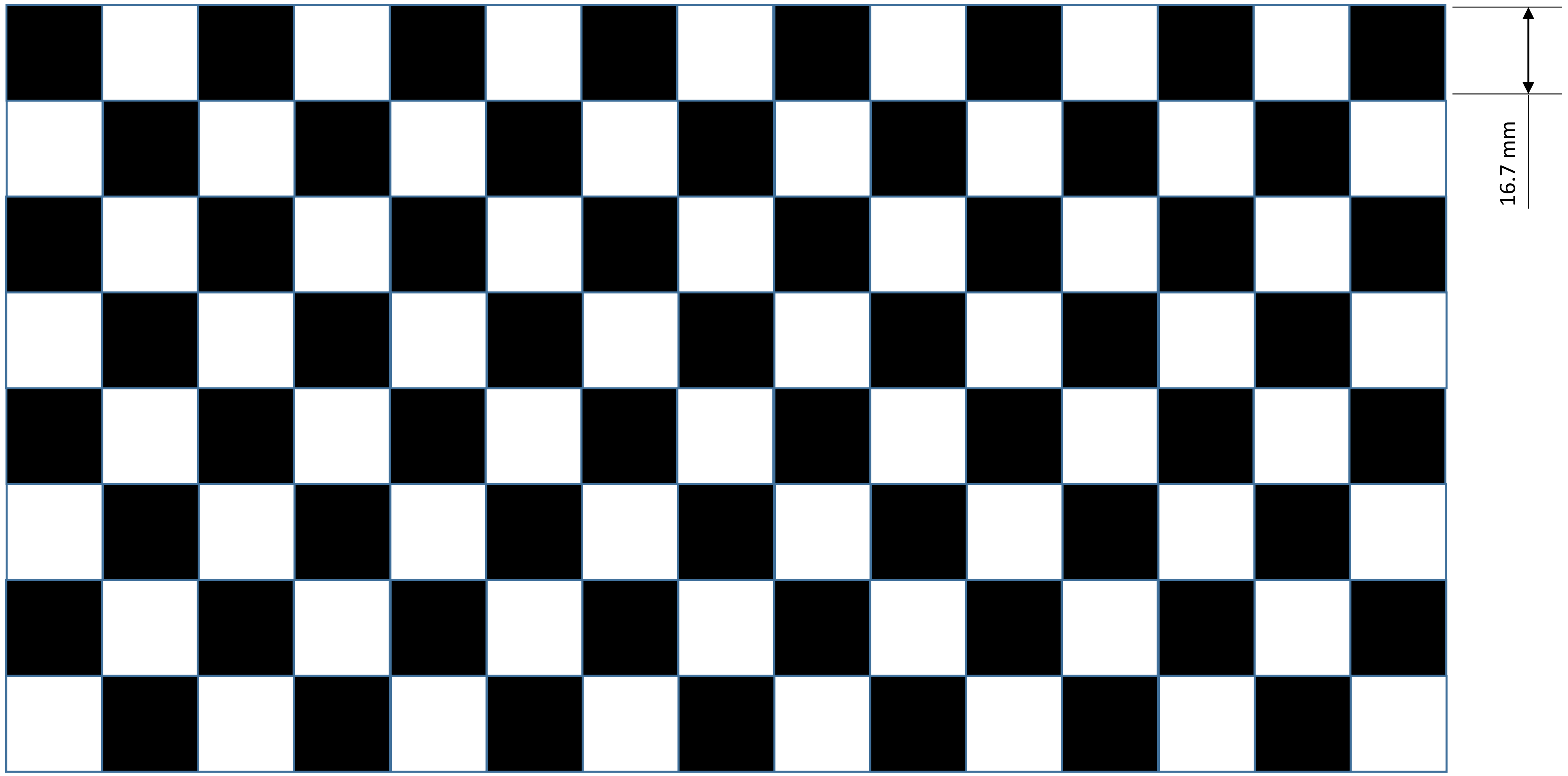


The workflow to calibrate your camera using the app:

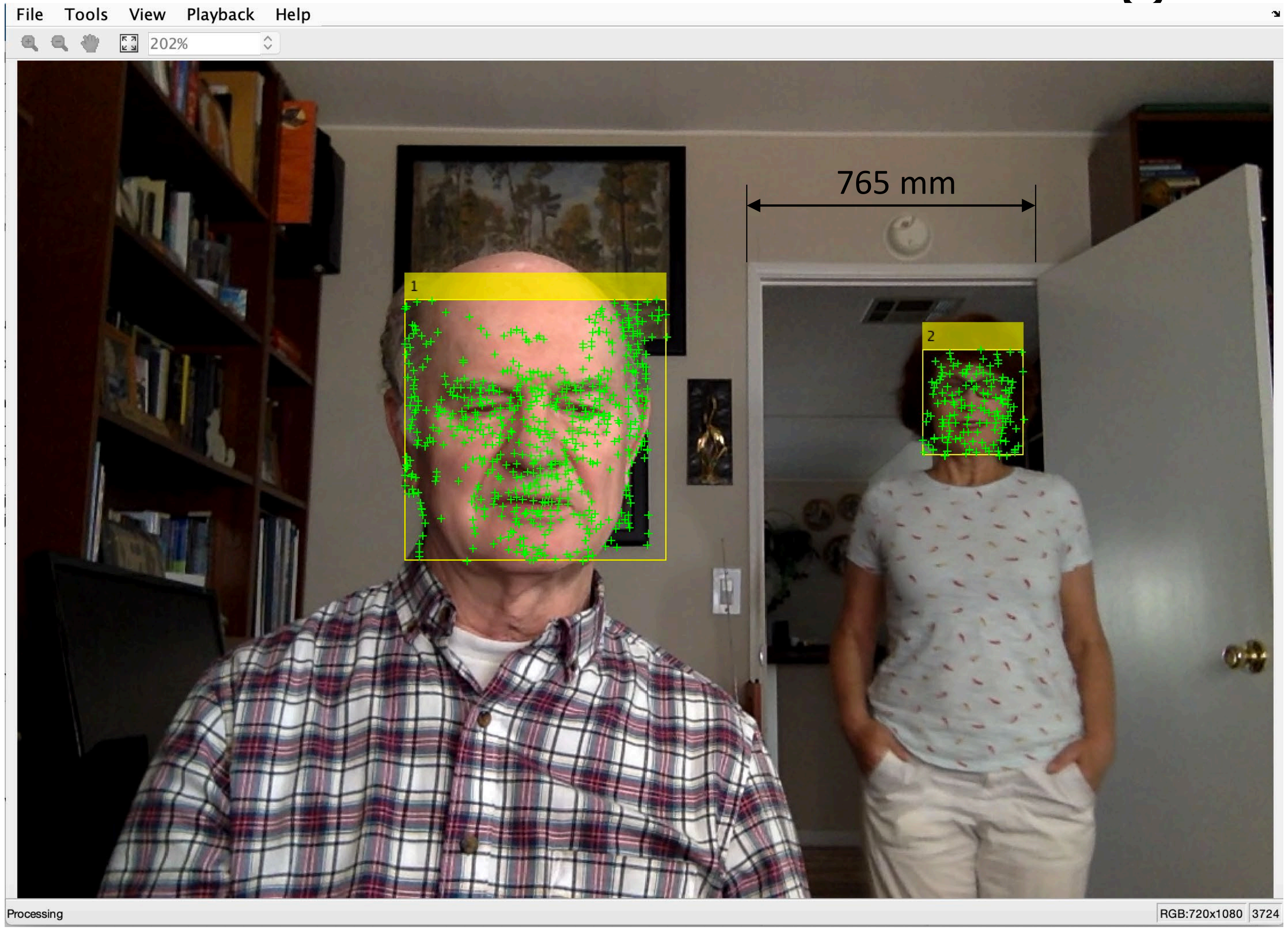
1. Prepare images, camera, and calibration pattern.
2. Add images and select standard or fisheye camera model.
3. Calibrate the camera.
4. Evaluate calibration accuracy.
5. Adjust parameters to improve accuracy (if necessary).
6. Export the parameters object.

In some cases, the default values work well, and we do not need to make any improvements before exporting parameters. We can also make improvements using the camera calibration functions directly in the MATLAB workspace.

Calibration Checkerboard Pattern



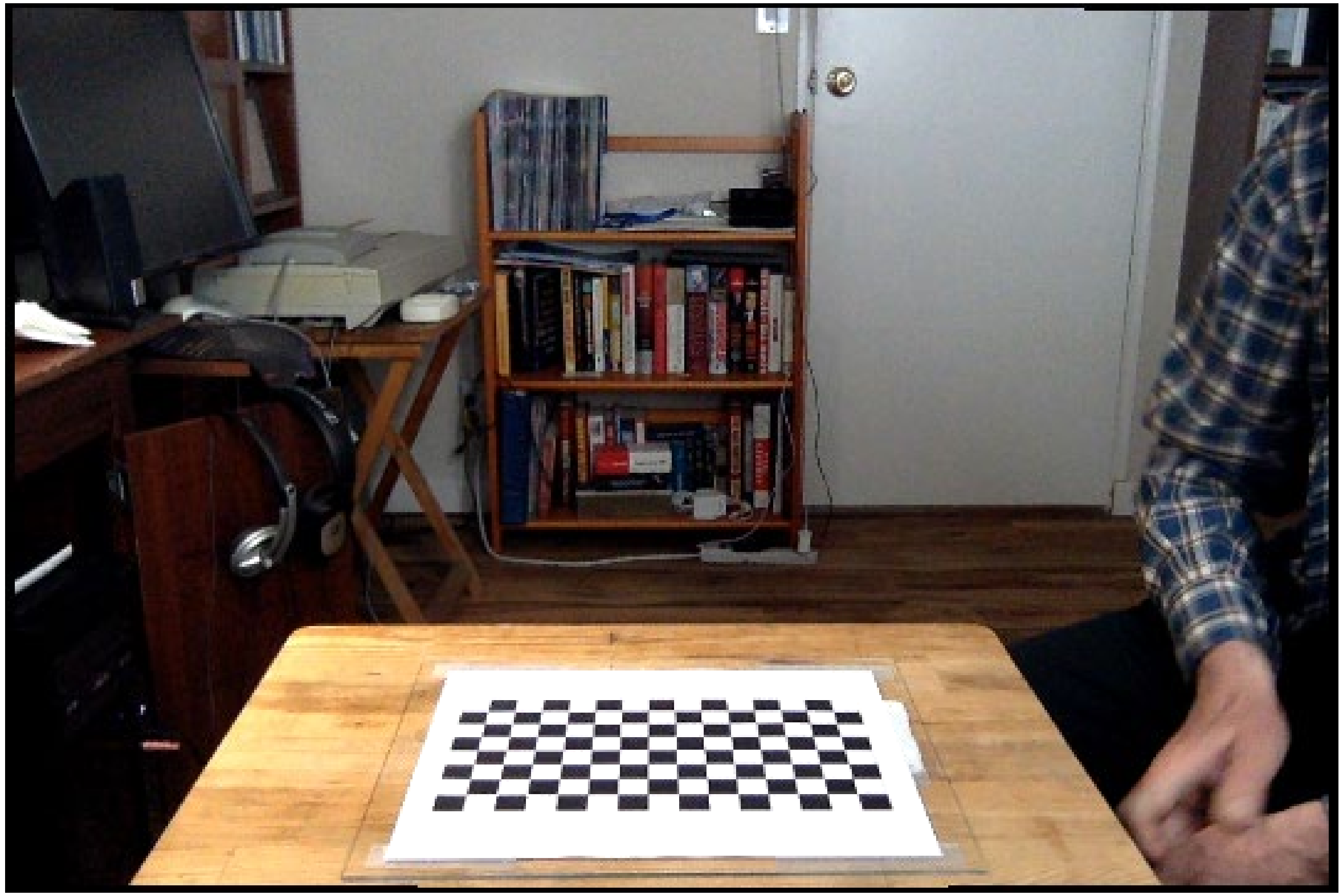
Smart Track Detection and Tracking Demo 1



Ground truth

1. Door width: 765 mm
2. Distance from the camera to the door: 3100 mm

Camera Calibration for Depth Measurement

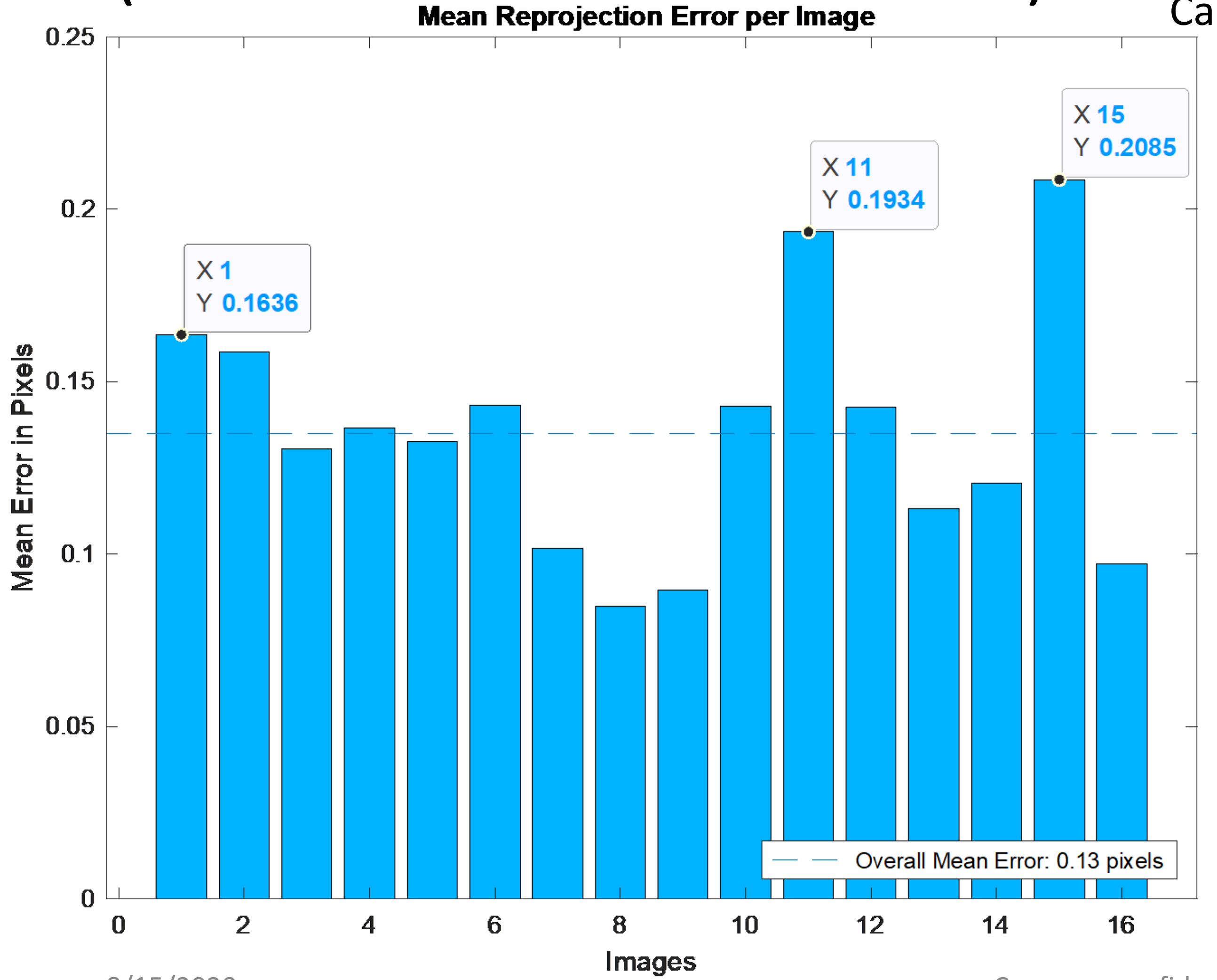


An advanced camera calibration technique and and improved calibration procedure [1] enables depth perception and measurement of exact relative position of objects in a 3-D scene. This procedure uses pictures (at least one picture in the set) of the calibration checkerboard pattern placed on a horizontal surface and at various distances from the camera.

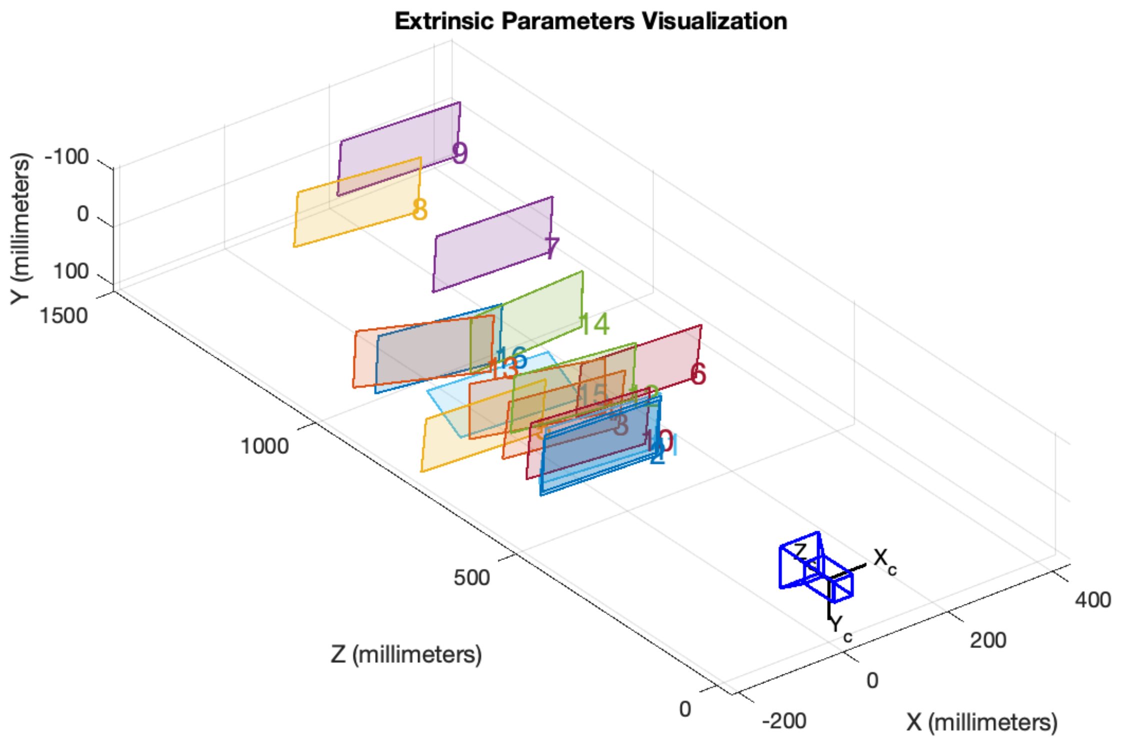
Reference

[1] Z. Zhang. “A flexible new technique for camera calibration”. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330-1334, 2000.

MacBook Camera Calibration Results (Modified Procedure)



Calibration results obtained with cropped 1080x720 images



MacBook Camera Calibration Data (Improved Procedure)

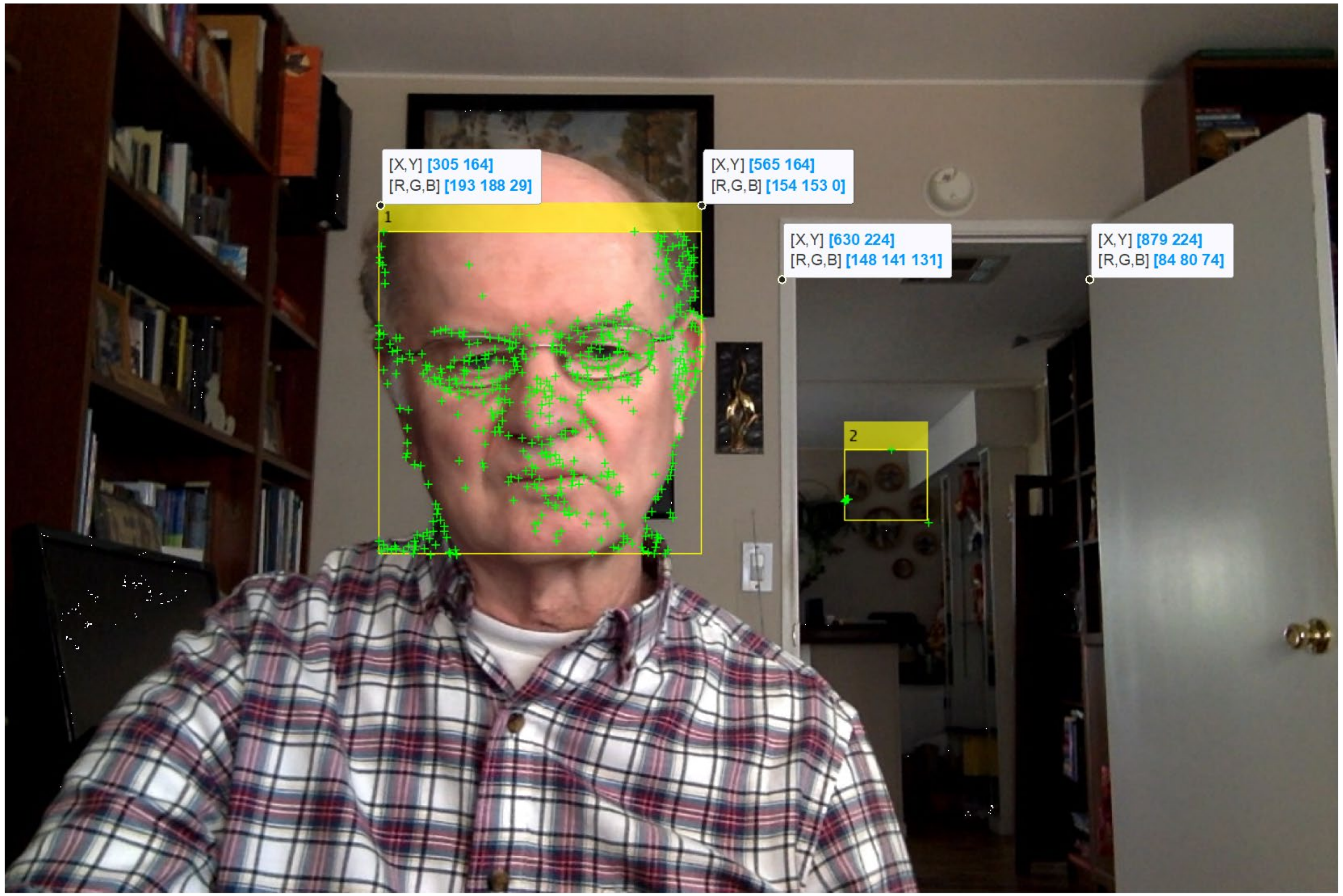
Calibration data generated with cropped 1080x720 images

```
                    Standard Errors of Estimated Camera Parameters
                    ----------------------------------------------------------------
Intrinsic parameters
----------
Focal length (pixels):     [ 1030.3006 +/- 3.1446     1026.6241 +/- 3.1596  ]
Principal point (pixels): [  565.8448 +/- 0.9619      351.7572 +/- 1.2309  ]
Radial distortion:         [    0.1107 +/- 0.0029       -0.3116 +/- 0.0136  ]

Extrinsic parameters
----------
Rotation vectors:
              [   -0.0388 +/- 0.0012       0.0806 +/- 0.0020      -3.1204 +/- 0.0001  ]
              [   -0.0283 +/- 0.0011       0.0982 +/- 0.0019      -3.1171 +/- 0.0001  ]
              [   -0.0517 +/- 0.0015       0.0922 +/- 0.0025      -3.0593 +/- 0.0001  ]
              [    0.5028 +/- 0.0015      -0.0896 +/- 0.0020       3.0686 +/- 0.0003  ]
              [    0.1142 +/- 0.0015      -0.2143 +/- 0.0016       3.1228 +/- 0.0002  ]
              [    0.1155 +/- 0.0026      -0.2415 +/- 0.0017       3.1233 +/- 0.0003  ]
              [   -0.0490 +/- 0.0033       0.1062 +/- 0.0034      -3.1258 +/- 0.0002  ]
              [   -0.1881 +/- 0.0053       0.1422 +/- 0.0064      -3.1254 +/- 0.0003  ]
              [    0.1043 +/- 0.0043      -0.1534 +/- 0.0043       3.1342 +/- 0.0005  ]
              [   -0.1025 +/- 0.0013       0.1084 +/- 0.0020      -3.1051 +/- 0.0001  ]
              [   -0.0233 +/- 0.0013       0.1533 +/- 0.0017      -3.0789 +/- 0.0001  ]
              [   -0.1469 +/- 0.0014       0.0695 +/- 0.0022      -3.1039 +/- 0.0002  ]
              [   -0.5822 +/- 0.0015       0.0779 +/- 0.0015      -3.0645 +/- 0.0004  ]
              [   -0.0412 +/- 0.0025      -0.0686 +/- 0.0033       3.1230 +/- 0.0002  ]
              [   -0.1167 +/- 0.0013      -1.7564 +/- 0.0017      -2.5449 +/- 0.0014  ]
              [   -0.1994 +/- 0.0023       0.0472 +/- 0.0028      -3.1055 +/- 0.0002  ]
```

```
Translation vectors (millimeters):
       [   96.4046 +/- 0.5143      64.5885 +/- 0.6546      550.5212 +/- 1.6995  ]
       [   90.4590 +/- 0.5107      68.5124 +/- 0.6508      548.5164 +/- 1.6895  ]
       [   98.6175 +/- 0.6094      68.4383 +/- 0.7744      650.9612 +/- 2.0103  ]
       [  106.6916 +/- 0.6631      67.0325 +/- 0.8317      695.2054 +/- 2.1673  ]
       [   12.7139 +/- 0.6838      83.4558 +/- 0.8648      731.5940 +/- 2.2187  ]
       [  304.0288 +/- 0.6923      87.8266 +/- 0.8512      726.6687 +/- 2.2589  ]
       [  332.1909 +/- 1.0661      63.0841 +/- 1.3365     1127.6549 +/- 3.4868  ]
       [  271.6598 +/- 1.3133      86.0429 +/- 1.6424     1376.8447 +/- 4.2820  ]
       [  432.5584 +/- 1.4155      96.4020 +/- 1.7608     1488.5867 +/- 4.6116  ]
       [  107.8054 +/- 0.5620      74.9055 +/- 0.7085      596.1518 +/- 1.8423  ]
       [  146.6130 +/- 0.5824     110.6384 +/- 0.7351      626.2682 +/- 1.9201  ]
       [   97.5076 +/- 0.5837       2.9330 +/- 0.7397      615.6807 +/- 1.9087  ]
       [  -11.0190 +/- 0.7750      12.1724 +/- 0.9918      825.5611 +/- 2.5681  ]
       [  190.1557 +/- 0.7995      21.4698 +/- 1.0292      861.9367 +/- 2.6344  ]
       [  141.2601 +/- 0.7453     100.7081 +/- 0.9397      799.1781 +/- 2.5892  ]
       [   63.6976 +/- 0.8503      55.5599 +/- 1.0806      902.0536 +/- 2.8046  ]
```

Smart Track Demo 1, Case 1: Ground Truth

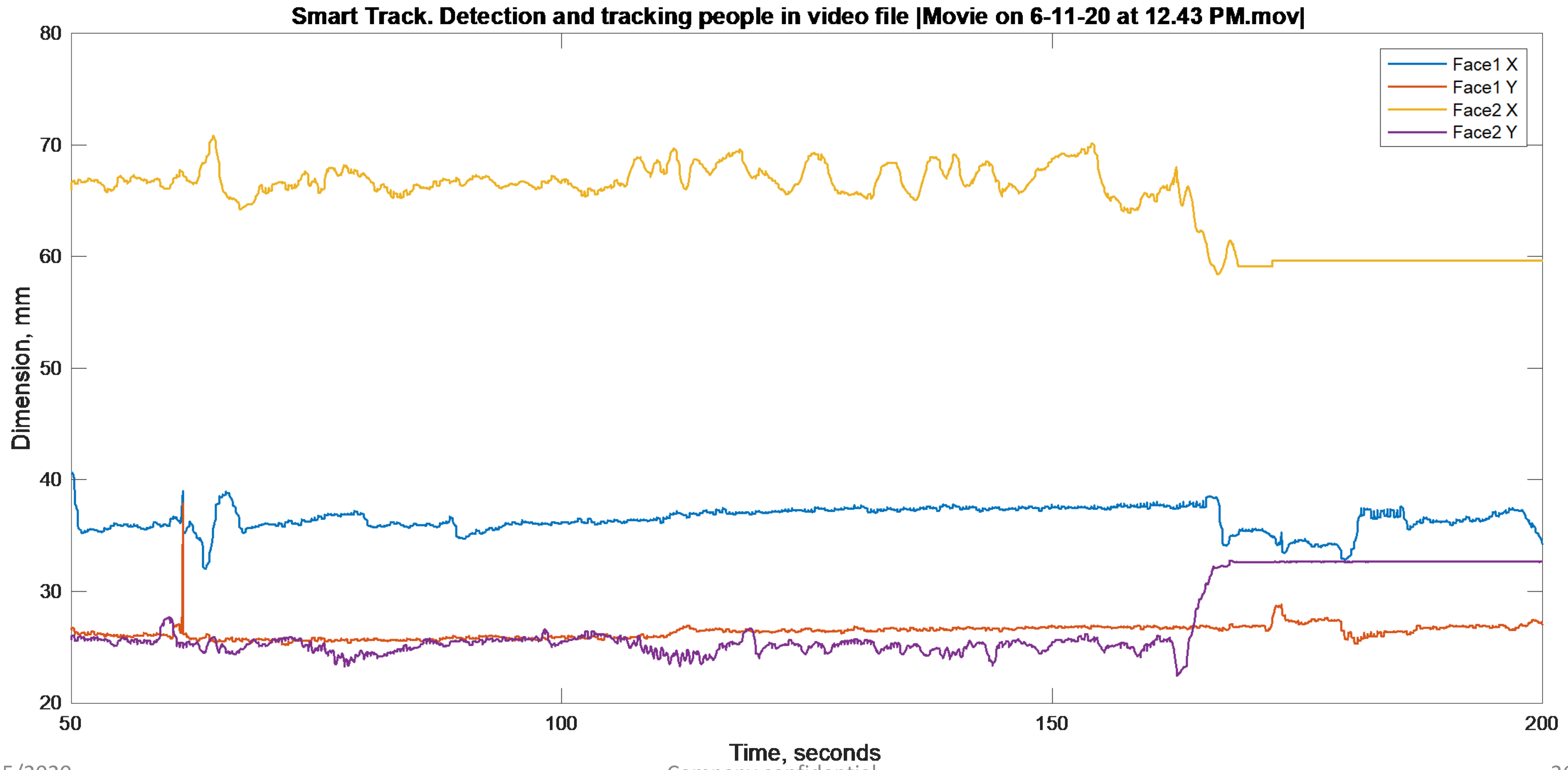


Ground truth

1. Door width: 765 mm
2. Distance from the camera to the door: 3100 mm

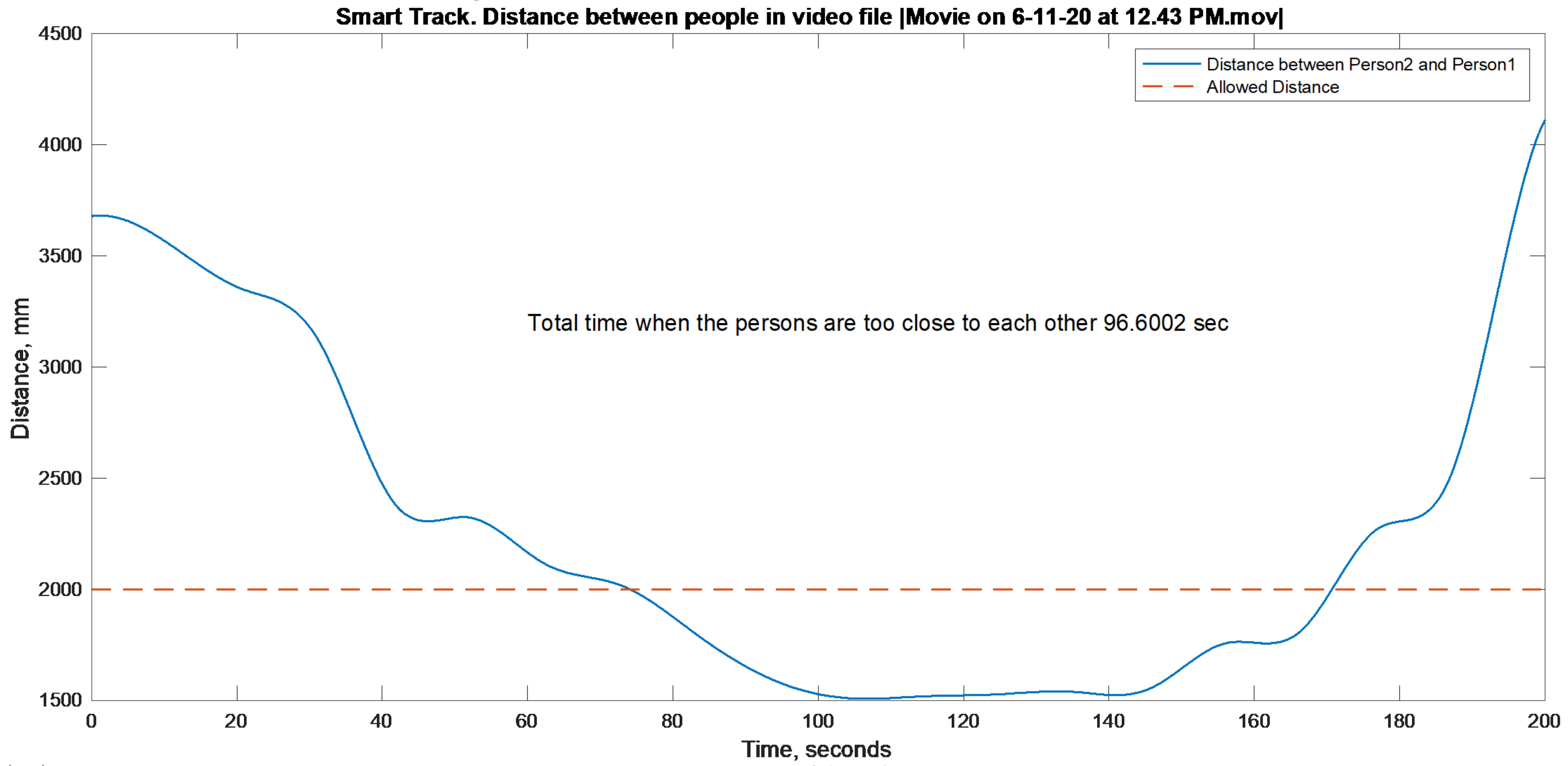
Detection and Tracking People in Video (X,Y)

After camera calibration and using Ground Truth



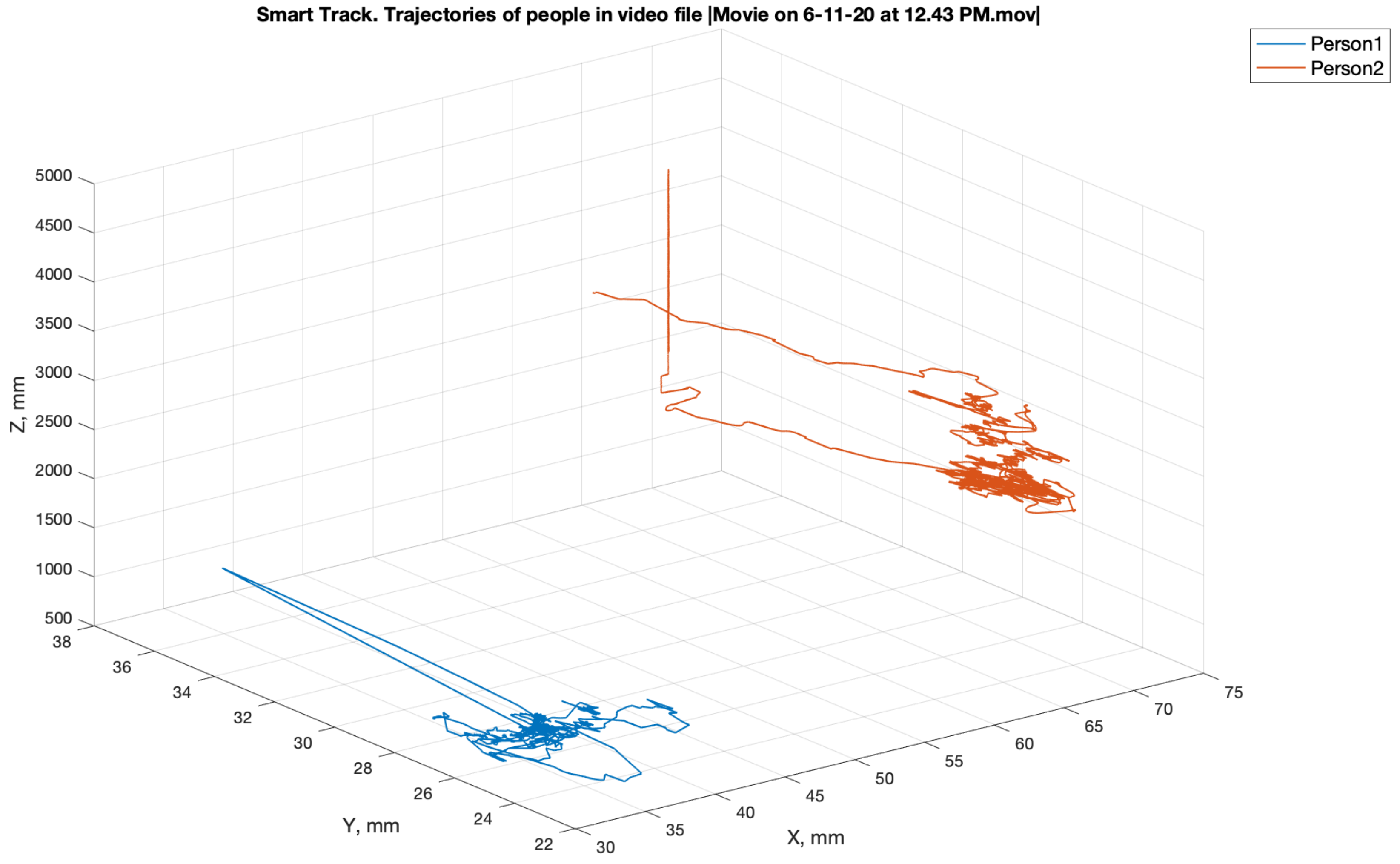
Distance Between People in Video (X, Y, Z)

After camera calibration and using Ground Truth

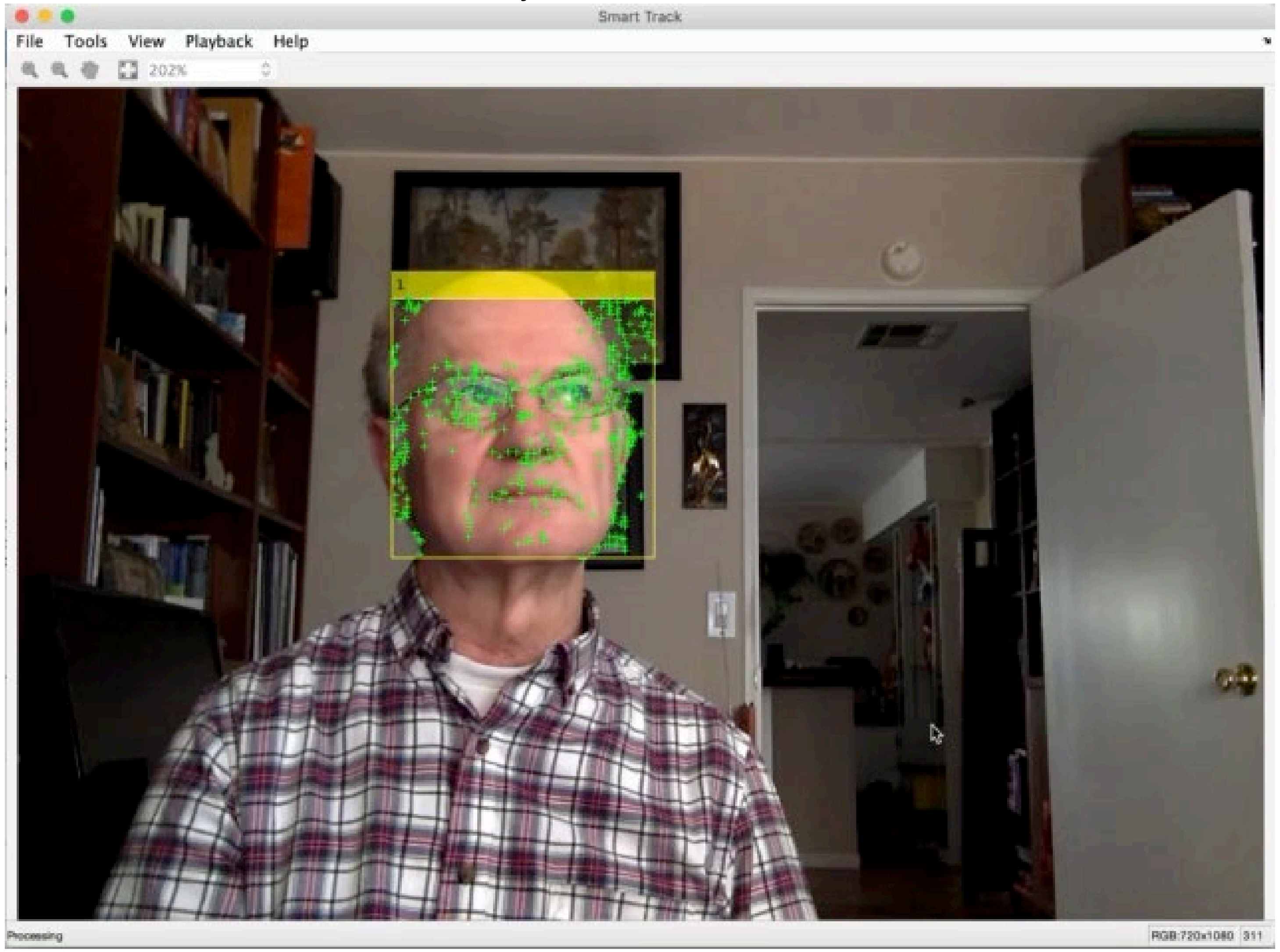


Trajectories (Traces) of People in Video (3-D)

After camera calibration and using Ground Truth



Smart Track Demo 1, Case 1 Recorded Video



Smart Track Demo 1, Case 2

This part of Demo 1 performs automatic detection and motion-based tracking of moving objects in a video from a stationary camera. The problem of motion-based object tracking can be divided into two parts:

1. Detecting moving objects in each frame
2. Associating the detections corresponding to the same object over time

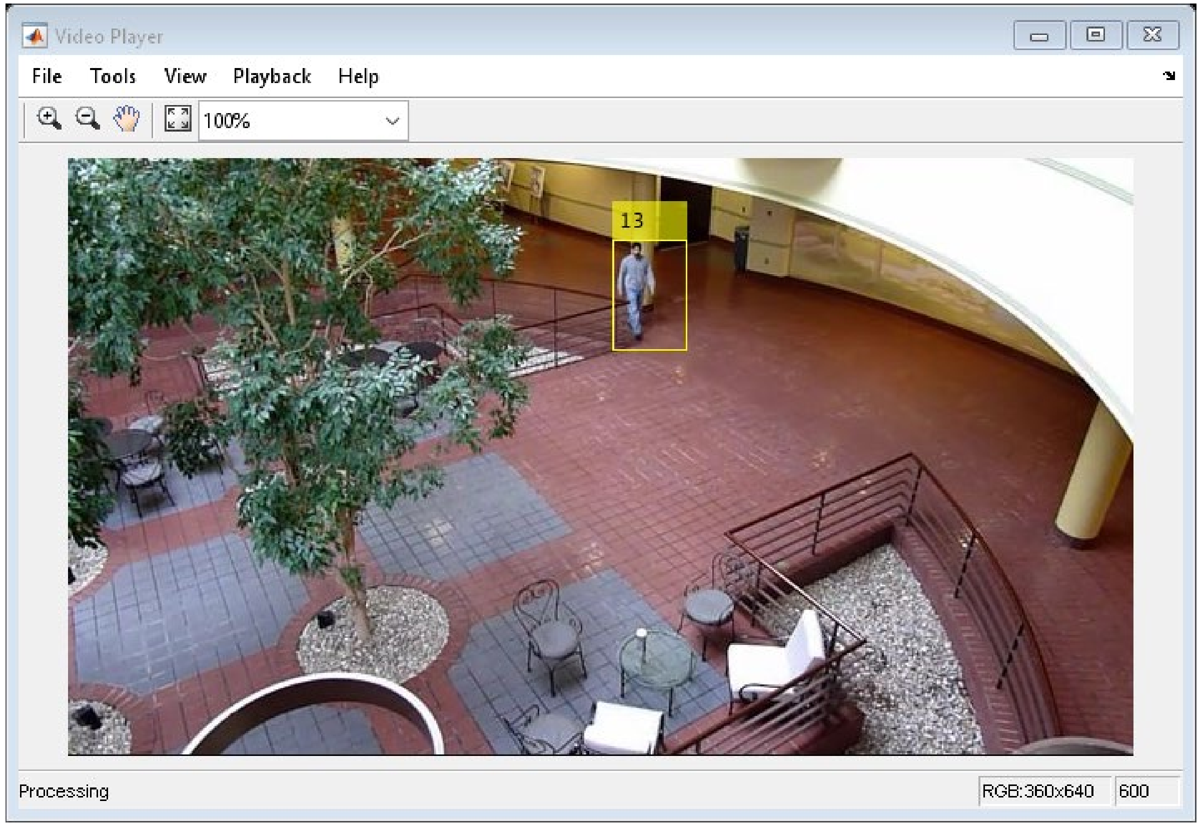
The detection of moving objects uses a background subtraction algorithm based on Gaussian mixture models. Morphological operations are applied to the resulting foreground mask to eliminate noise. Finally, blob analysis detects groups of connected pixels, which are likely to correspond to moving objects.

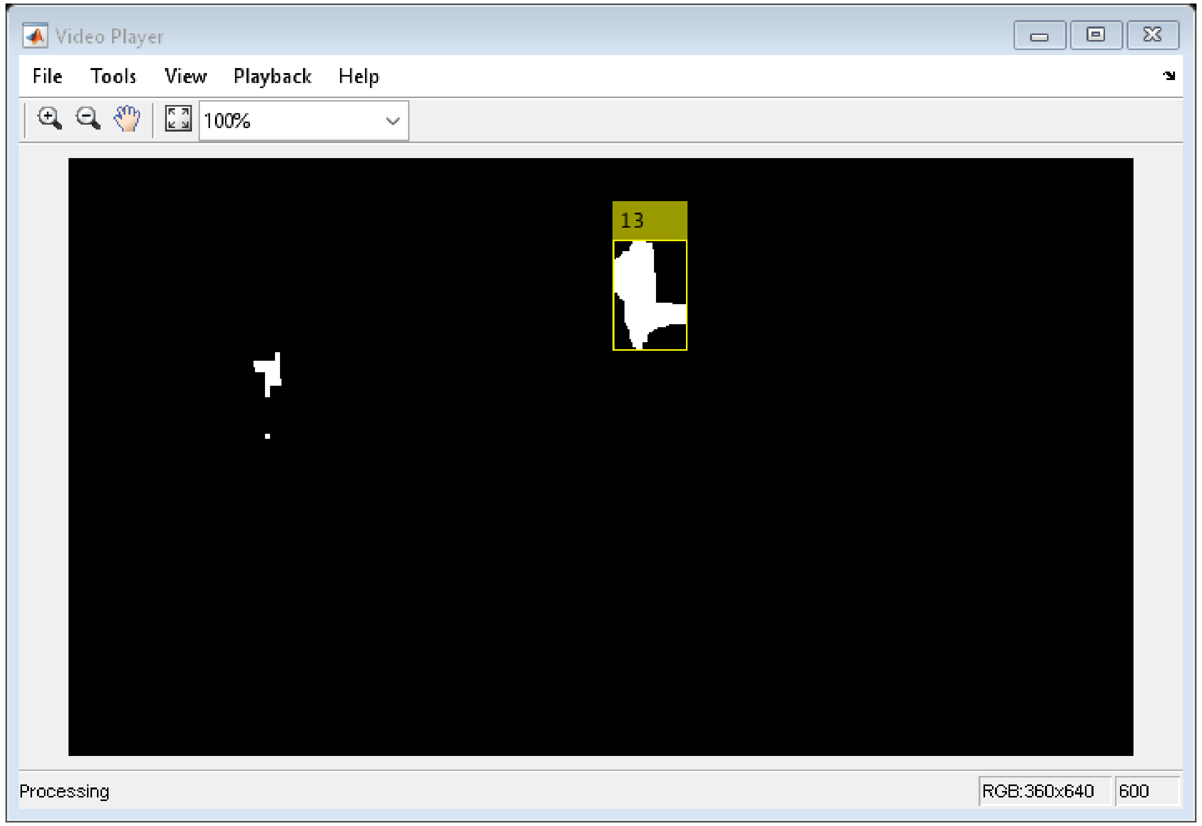
The association of detections to the same object is based solely on motion. The motion of each track is estimated by a Kalman filter. The filter is used to predict the track's location in each frame, and determine the likelihood of each detection being assigned to each track.

Track maintenance becomes an important aspect of this app. In any given frame, some detections may be assigned to tracks, while other detections and tracks may remain unassigned. The assigned tracks are updated using the corresponding detections. The unassigned tracks are marked invisible. An unassigned detection begins a new track.

Each track keeps count of the number of consecutive frames, where it remained unassigned. If the count exceeds a specified threshold, the example assumes that the object left the field of view and it deletes the track.

Detection and Tracking of Moving People. Demo1, Case 2





Detection of moving objects and tracking them across video frames

DMGsmart

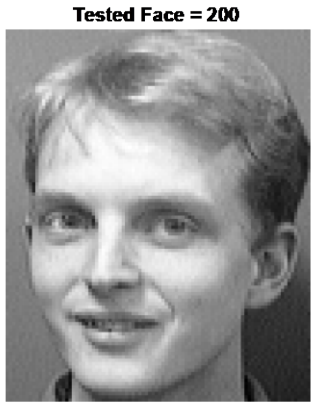
MATLAB Face Recognition Code for DMG Smart Track

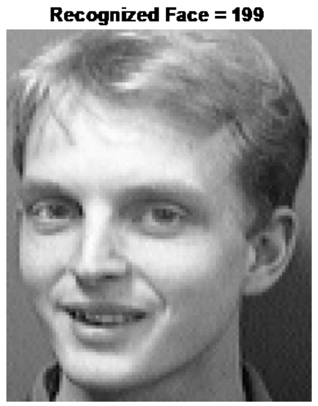
DMGSMAR-28: Write MATLAB code to integrate the modified functions and scrips into the PDT – Part 2

Face Recognition for DMG Smart Track

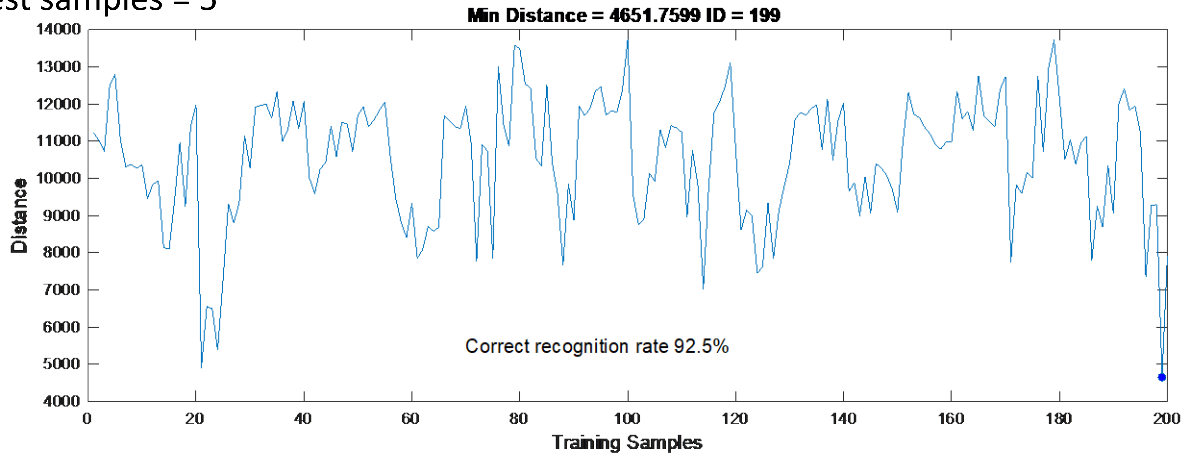
1. Algorithms for face recognition typically extract facial features and compare them to a database to find the best match. Face recognition is an important part of many biometric, security, and surveillance systems, as well as image and video indexing systems.
2. Face recognition leverages computer vision to extract discriminative information from facial images, and pattern recognition or machine learning techniques to model the appearance of faces and to classify them.
3. MATLAB apps use computer vision techniques to perform feature extraction to encode the discriminative information required for face recognition as a compact feature vector using techniques and algorithms such as:
   - ❖ Dense local feature extraction with SURF, BRISK, or FREAK descriptors
   - ❖ Histogram of oriented gradients
   - ❖ Distance between detected facial landmarks such as eyes, noses, and lips
   - ❖ Machine learning techniques that can be applied to the extracted features to perform face recognition or classification using:
     - ➢ Supervised learning techniques such as support vector machines (SVM) and decision trees
     - ➢ Ensemble learning methods
     - ➢ Deep neural networks

Face Recognition Using 2-D Principal Component Analysis (2D PCA)





Number of classes = 40
Number of samples per class = 10
Number of training samples = 5
Number of test samples = 5



The 2D PCA, AKA the Karhunen–Loève transform, is a representation of a stochastic process as an infinite linear combination of orthogonal functions, analogous to a Fourier series representation of a function on a bounded interval. The transformation is also known as Hotelling trace transform and eigenvector transform, and is closely related to principal component analysis (PCA) technique widely used in image processing, image/data compression, and data analysis in many fields. However, the 2DPCA is not a very efficient pattern recognition approach as it cannot provide sufficiently high correct recognition rate for large number of classes and small number (1-2) of training samples per class.

Face Recognition Using Singular Value Decomposition (SVD)

Test Image

Recogized Image

In linear algebra, the singular value decomposition (SVD) is a factorization of a real or complex matrix that generalizes the eigen decomposition of a square normal matrix to any *m* x *n* matrix via an extension of the polar decomposition.

The SVD is applied extensively to the study of linear inverse problems and is useful in the analysis of regularization methods such as that of Tikhonov. It is widely used in statistics, where it is related to principal component analysis and to correspondence analysis, and in signal processing and pattern recognition. However, the SVD is not a very efficient pattern recognition approach as it does not deliver sufficiently high correct recognition rate for large number of classes and small number (1-2) of in training samples per class.

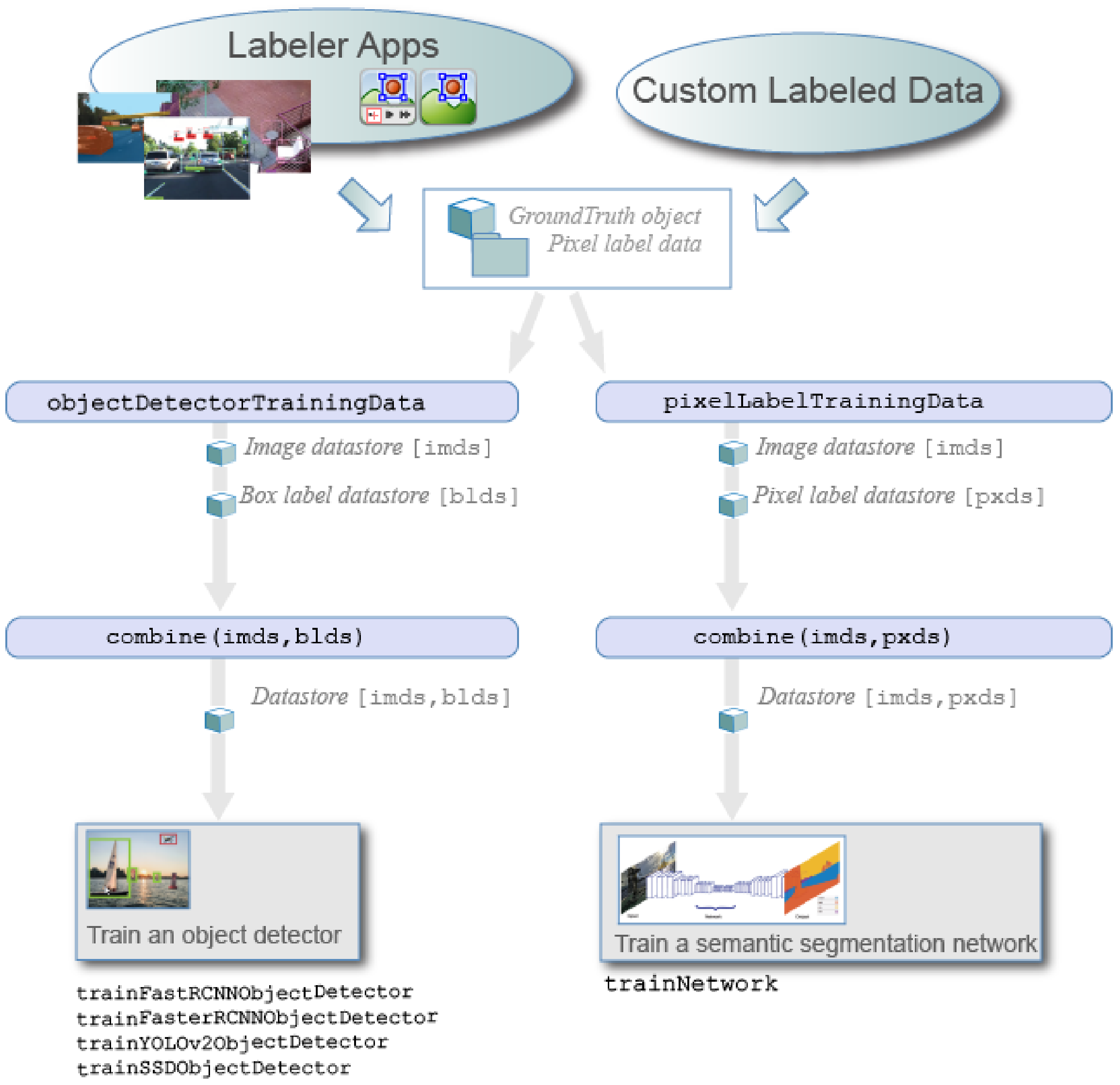
Advanced Face Recognition Techniques

Pattern recognition rule of thumb: often the quality of the training set is more important than the quality of the pattern recognition and classification algorithm

Deep learning architectures and semantic approach can create, modify, and analyze advanced face recognition classifier algorithms using **MATLAB labeling apps and visualization tools**.

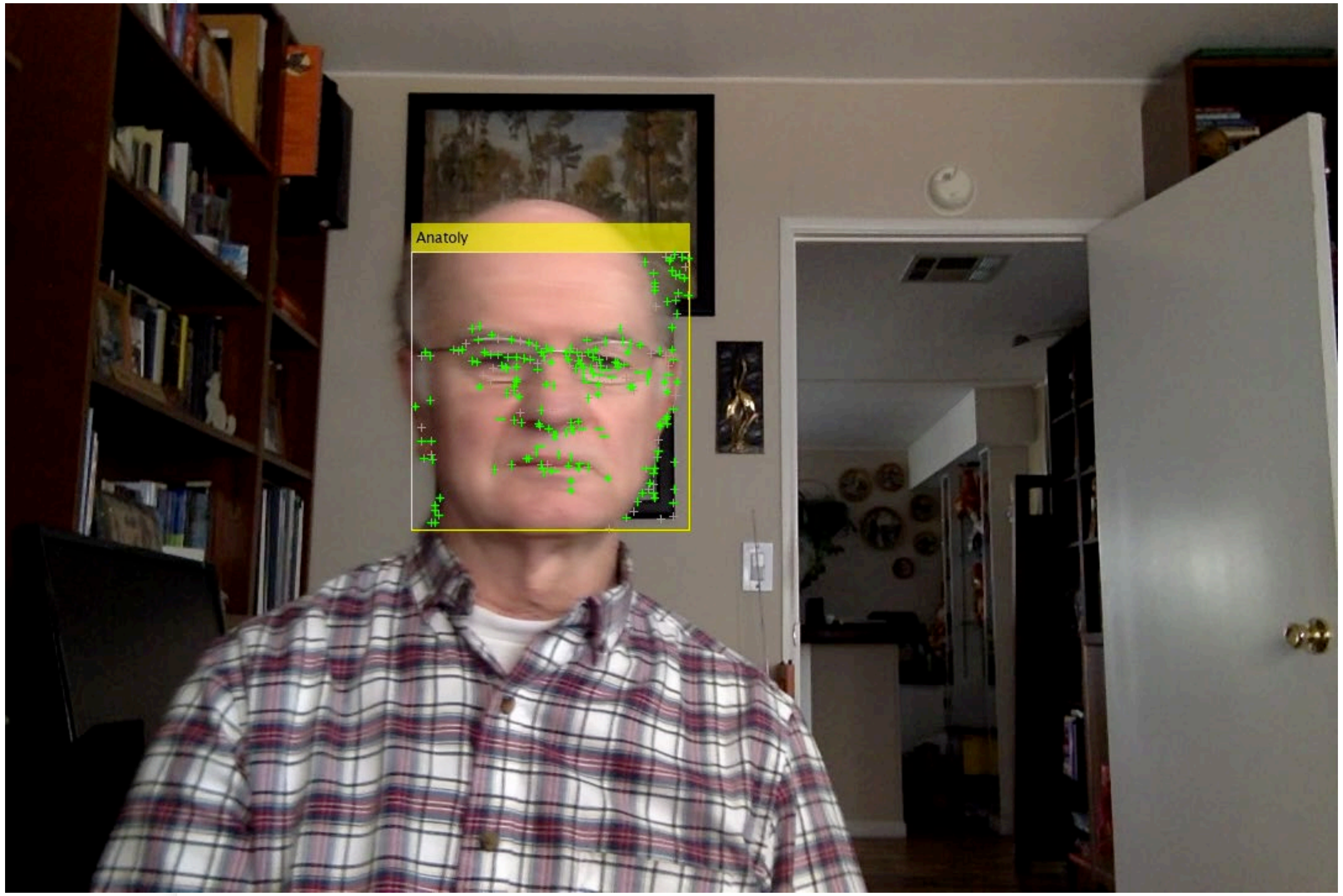
We use a MATLAB labeling app and Computer Vision Toolbox™ objects and functions to train classifier algorithms from image databases or ground truth data. Then we use the labeling app to interactively label people/faces and/or ground truth data in a video, image sequence, image collection, or custom data source. Finally, we use the labeled data to create training data to train an object detector and to train a semantic segmentation network to achieve an accurate face recognition.

Training Data for Object Detection, Semantic Segmentation, Recognition and Labeling

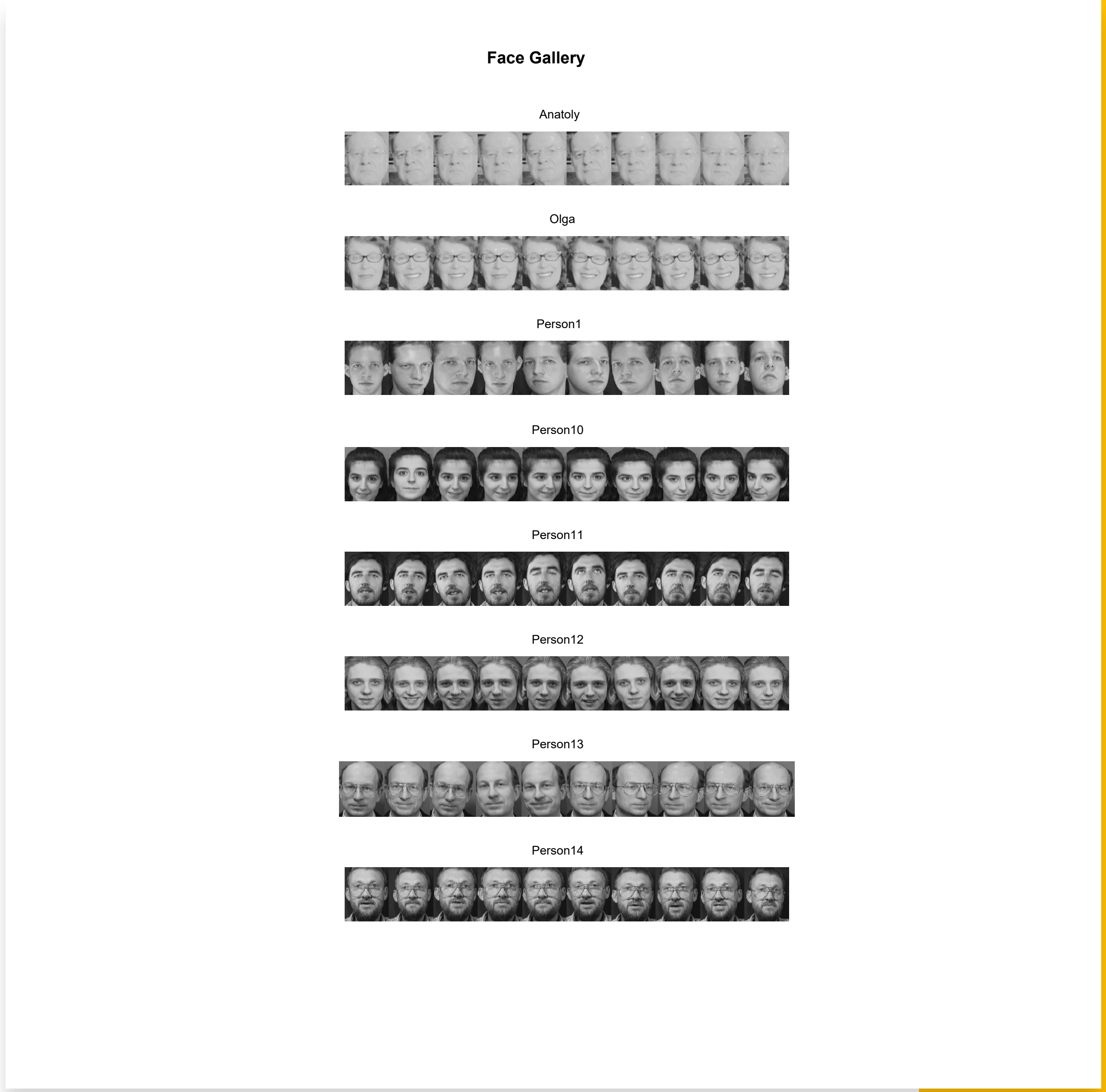


Smart Track Face Recognition, Labeling, and Tracking.

Correct Recognition Rate >99.8%

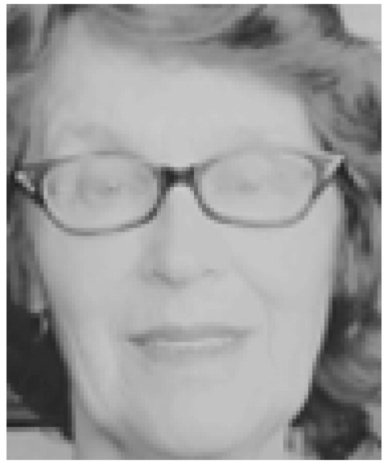


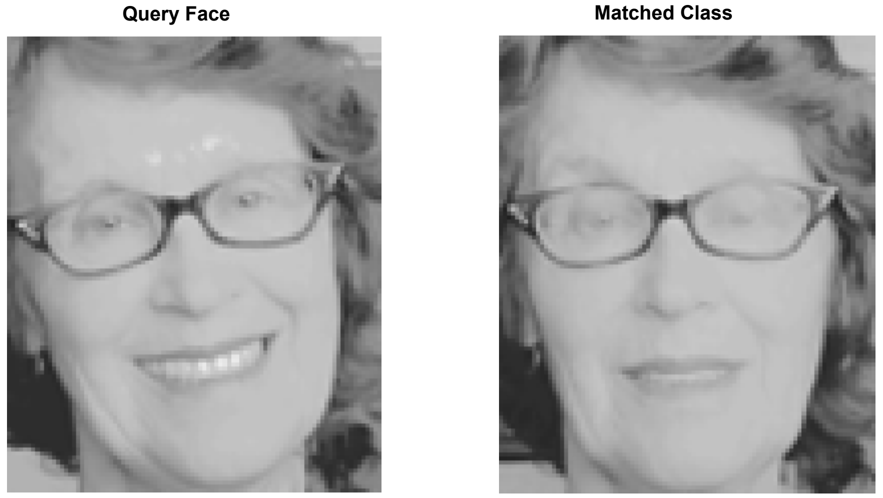
Creating and Testing a 40-Person Face Database



Face Recognition Results Using HOG Features

Histogram of oriented gradients (HOG) features provide satisfactory face recognition accuracy for a relatively small number of classes [1].

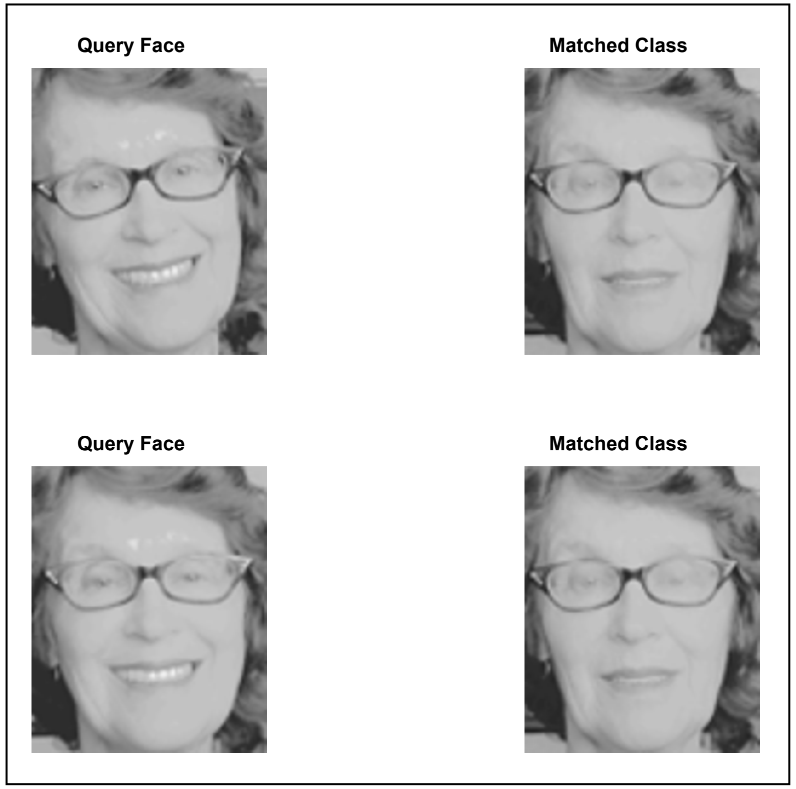




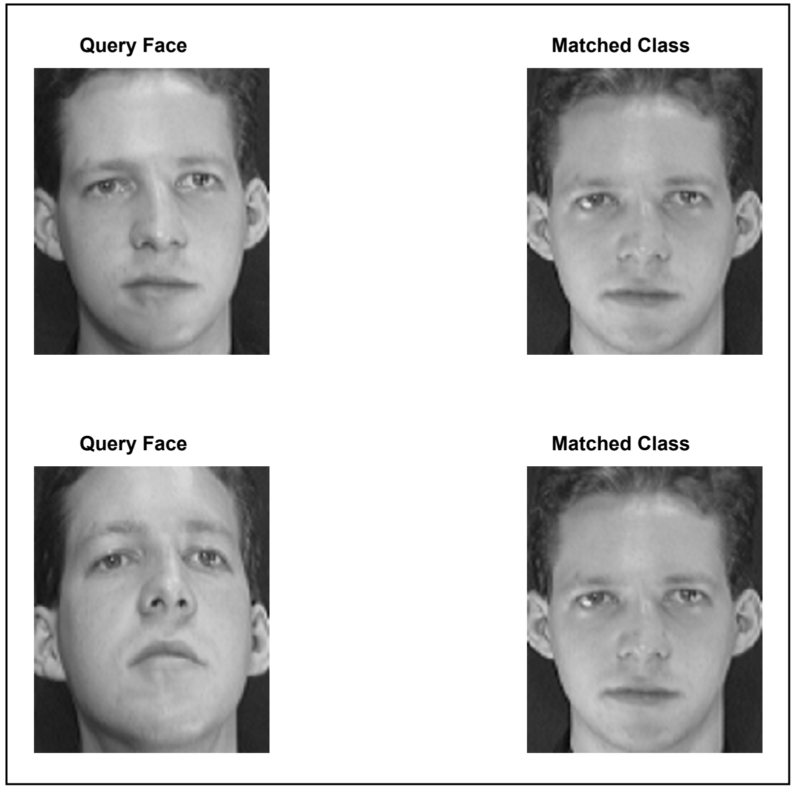
[1] N. Dalal and B. Triggs. "Histograms of Oriented Gradients for Human Detection", IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 1 (June 2005), pp. 886–893.

Face Recognition Accuracy with HOG Features

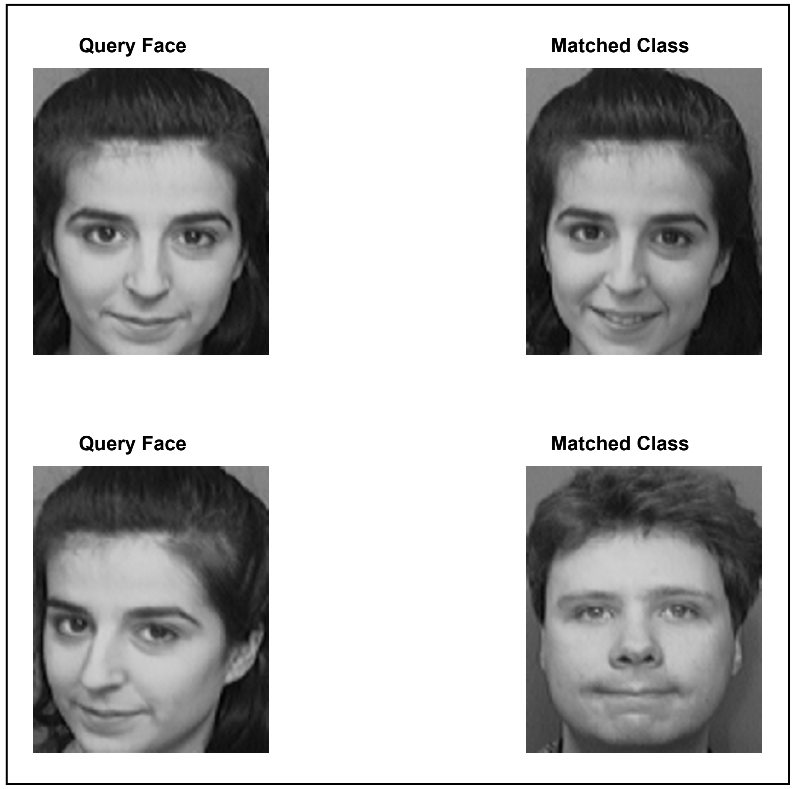
Correct Recognition



Correct Recognition



Incorrect Recognition



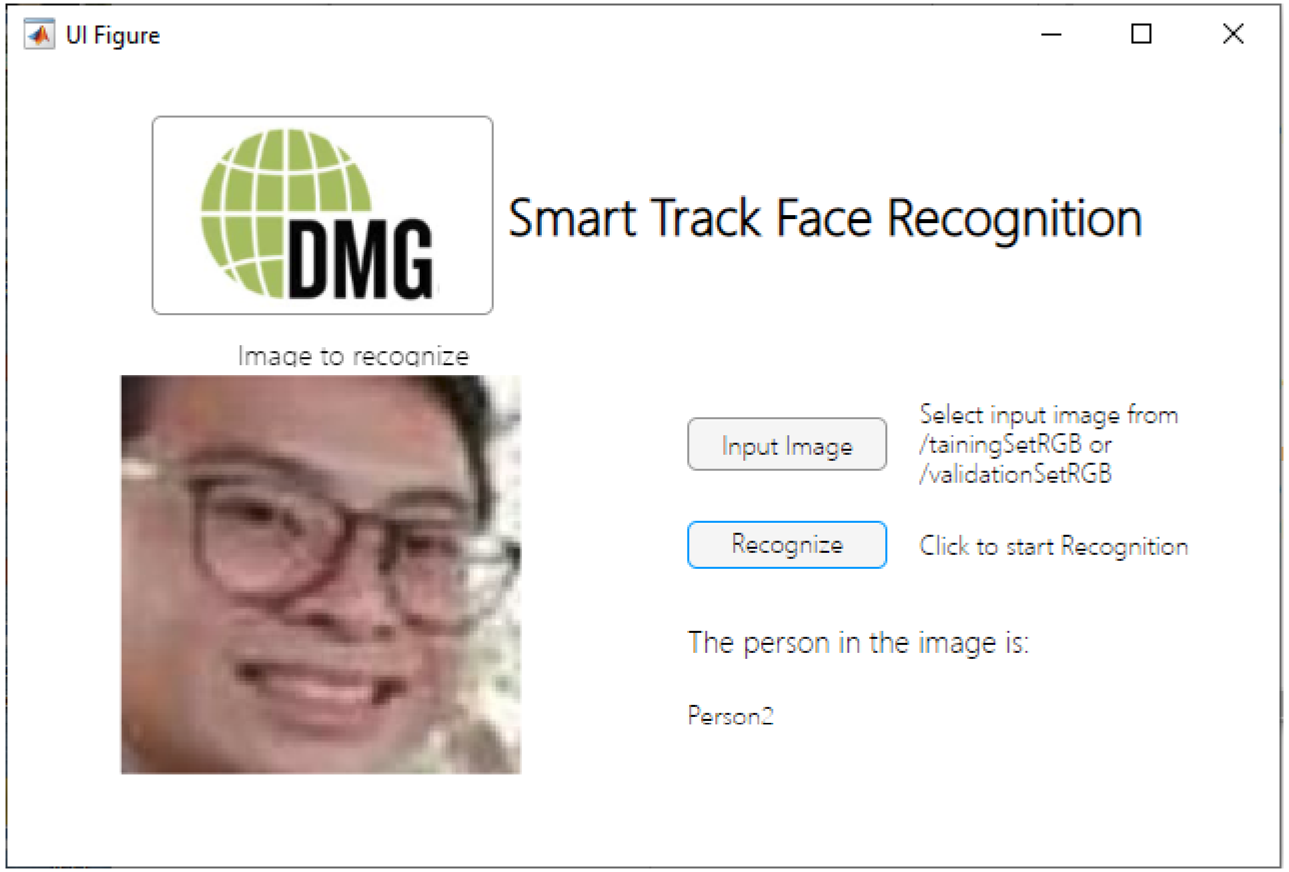
One incorrect face recognition out of 40 test trials. Correct recognition rate 97.5%

Face Recognition Testbed

Task 47. Scale Up existing face recognition model my adding up to 40 'labeled' classes

Deep Learning 25-Layer CNN with GUI

Training accuracy 100%, validation accuracy >66%. Correct ID


UI Figure

Smart Track Face Recognition

Image to recognize

Input Image

Select input image from /tainingSetRGB or /validationSetRGB

Recognize

Click to start Recognition

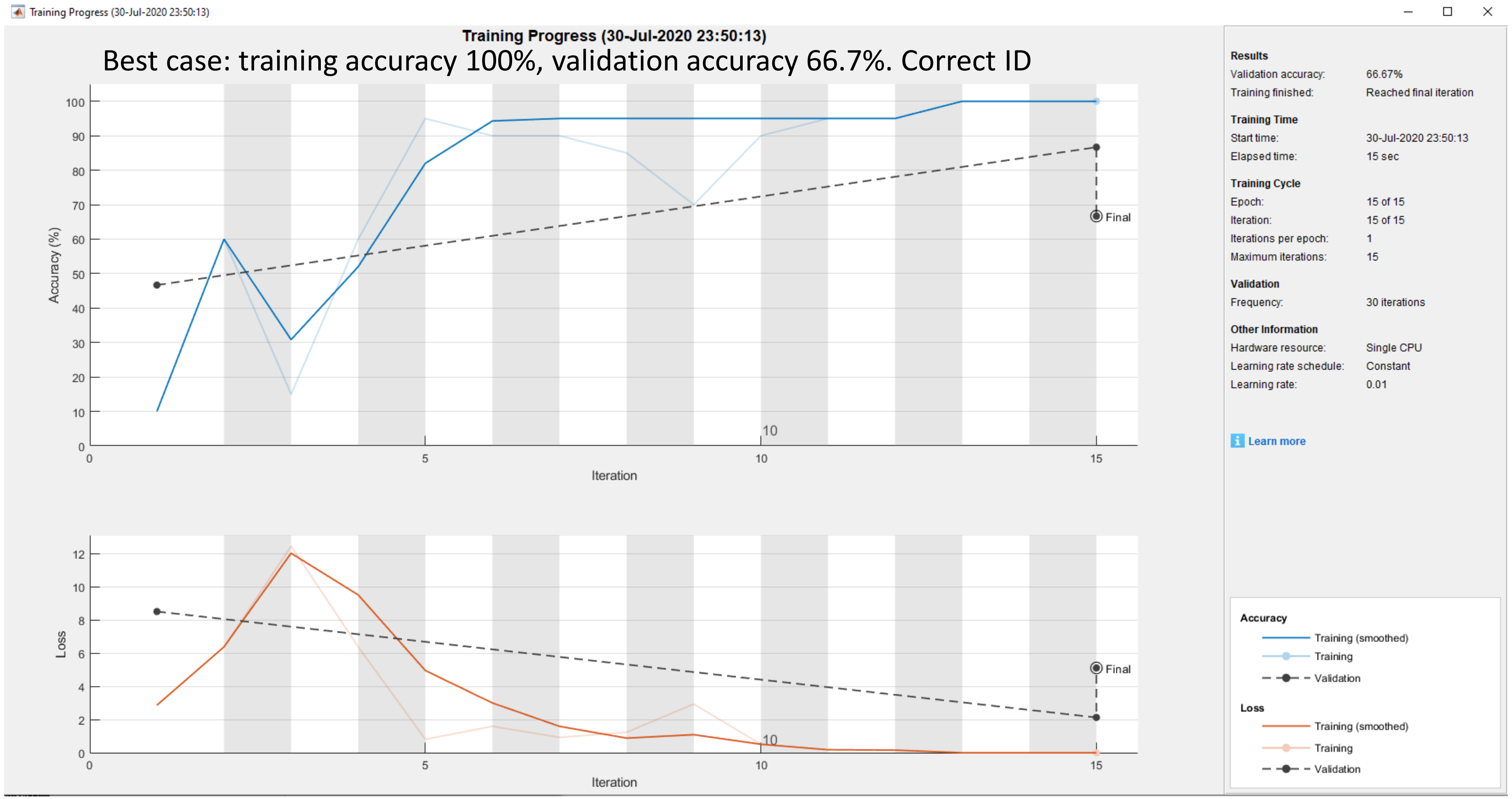
The person in the image is:

Person2

Face image database has 5 Persons/Classes
Training database has 4 images per class
Testing database has 3 images per class

Input image: Person1
Correct ID

25-Layer CNN Training, Testing, and Validation


Training Progress (30-Jul-2020 23:50:13)

Training Progress (30-Jul-2020 23:50:13)

Best case: training accuracy 100%, validation accuracy 66.7%. Correct ID

Results

Validation accuracy:	66.67%
Training finished:	Reached final iteration

Training Time

Start time:	30-Jul-2020 23:50:13
Elapsed time:	15 sec

Training Cycle

Epoch:	15 of 15
Iteration:	15 of 15
Iterations per epoch:	1
Maximum iterations:	15

Validation

Frequency:	30 iterations

Other Information

Hardware resource:	Single CPU
Learning rate schedule:	Constant
Learning rate:	0.01

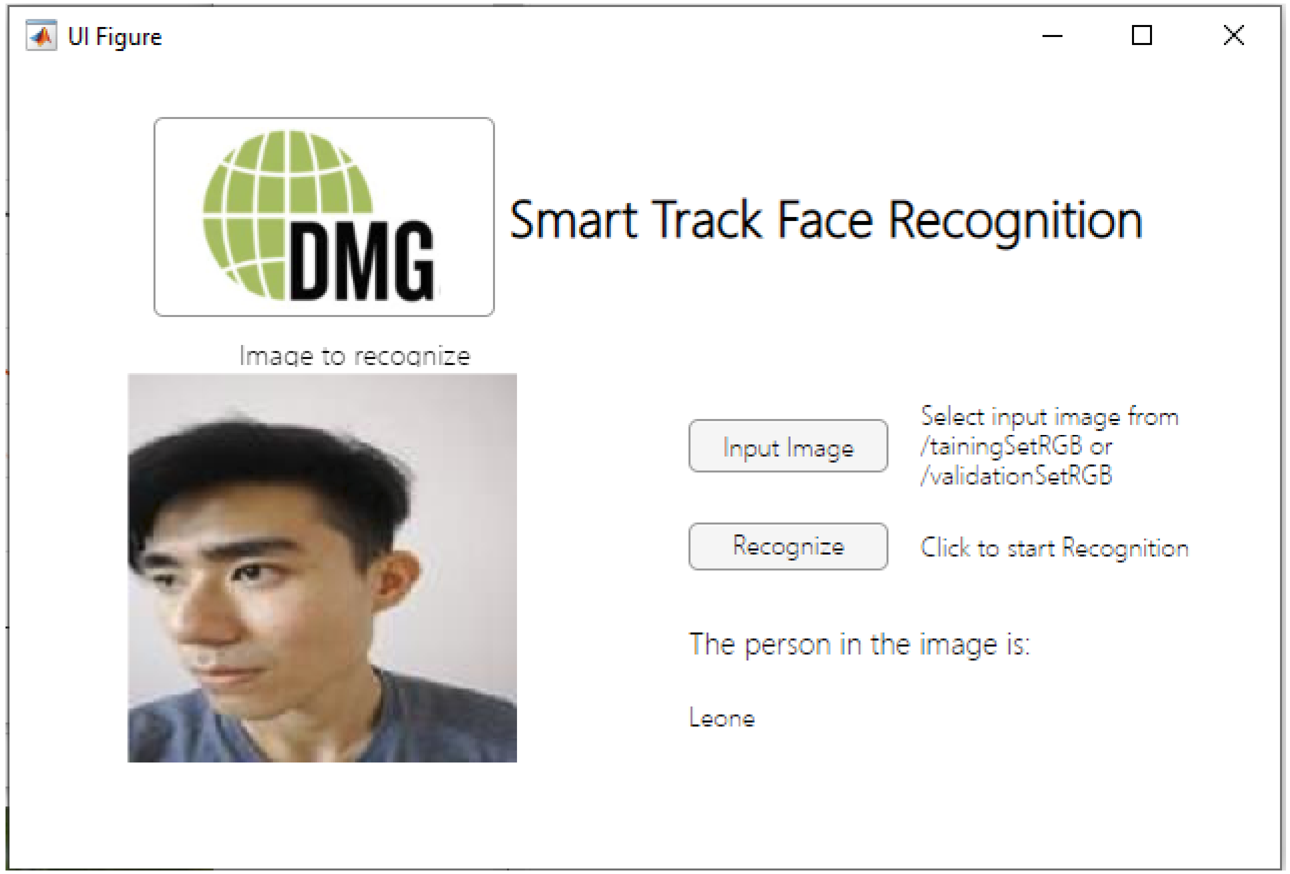
Learn more

Accuracy
- Training (smoothed)
- Training
- Validation

Loss
- Training (smoothed)
- Training
- Validation

Deep Learning 25-Layer CNN GUI. Wrong ID

Training accuracy 42%, validation accuracy 20%. Wrong ID



UI Figure

Smart Track Face Recognition

Image to recognize

Input Image

Select input image from /tainingSetRGB or /validationSetRGB

Recognize

Click to start Recognition

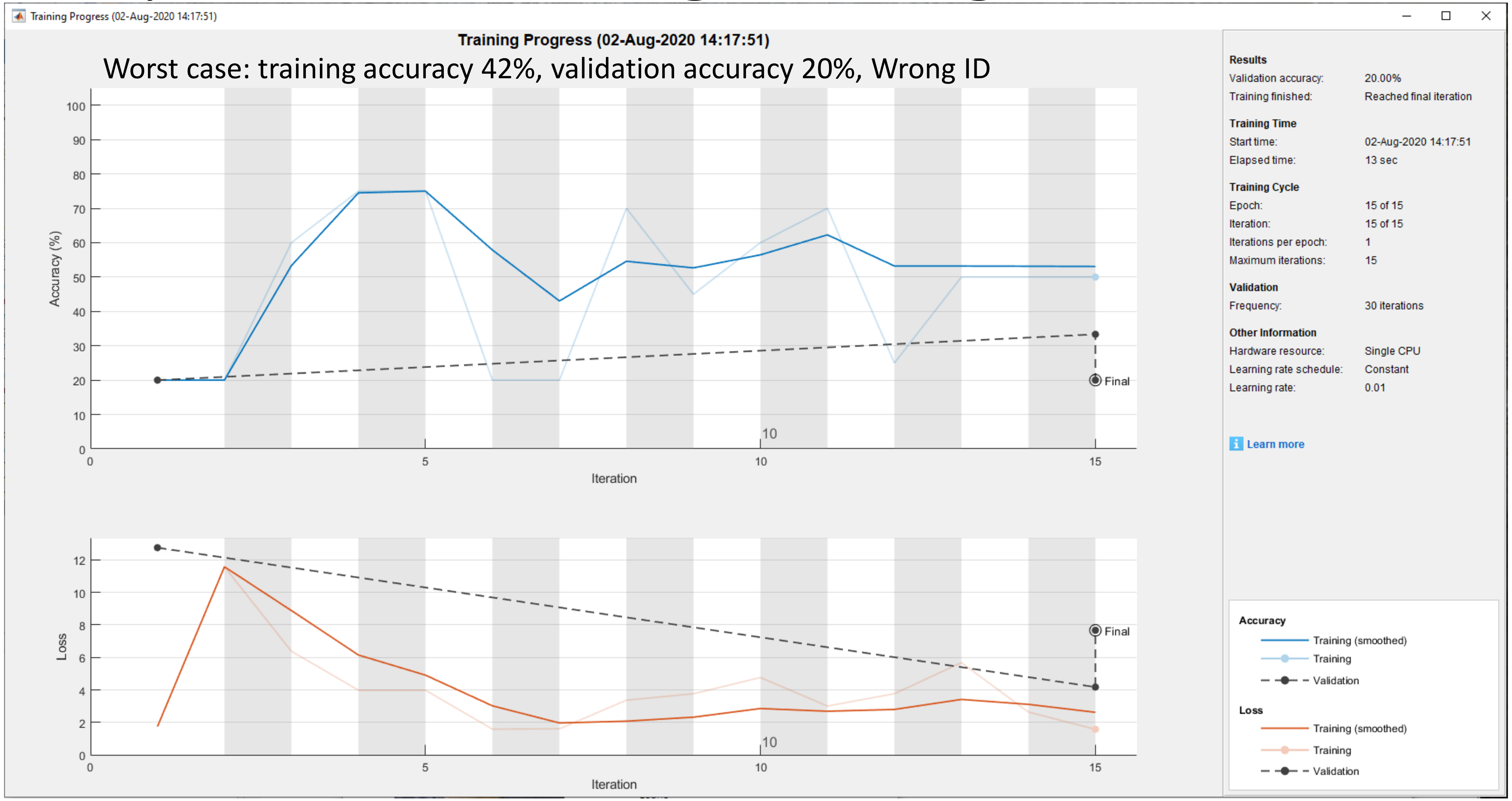
The person in the image is:

Leone

Face image database has 5 Persons/Classes
Training database has 4 images per class
Testing database has 3 images per class

Input image: Shawn
Recognized: Leone

25-Layer CNN Training, Testing, and Validation



Training Progress (02-Aug-2020 14:17:51)

Training Progress (02-Aug-2020 14:17:51)

Worst case: training accuracy 42%, validation accuracy 20%, Wrong ID

Results

Validation accuracy:	20.00%
Training finished:	Reached final iteration

Training Time

Start time:	02-Aug-2020 14:17:51
Elapsed time:	13 sec

Training Cycle

Epoch:	15 of 15
Iteration:	15 of 15
Iterations per epoch:	1
Maximum iterations:	15

Validation

Frequency:	30 iterations

Other Information

Hardware resource:	Single CPU
Learning rate schedule:	Constant
Learning rate:	0.01

Learn more

Accuracy
- Training (smoothed)
- Training
- Validation

Loss
- Training (smoothed)
- Training
- Validation

AlexNet CNN Face Detection and Recognition

Test image captured from a video, which is not in in either training or test/validation face databases

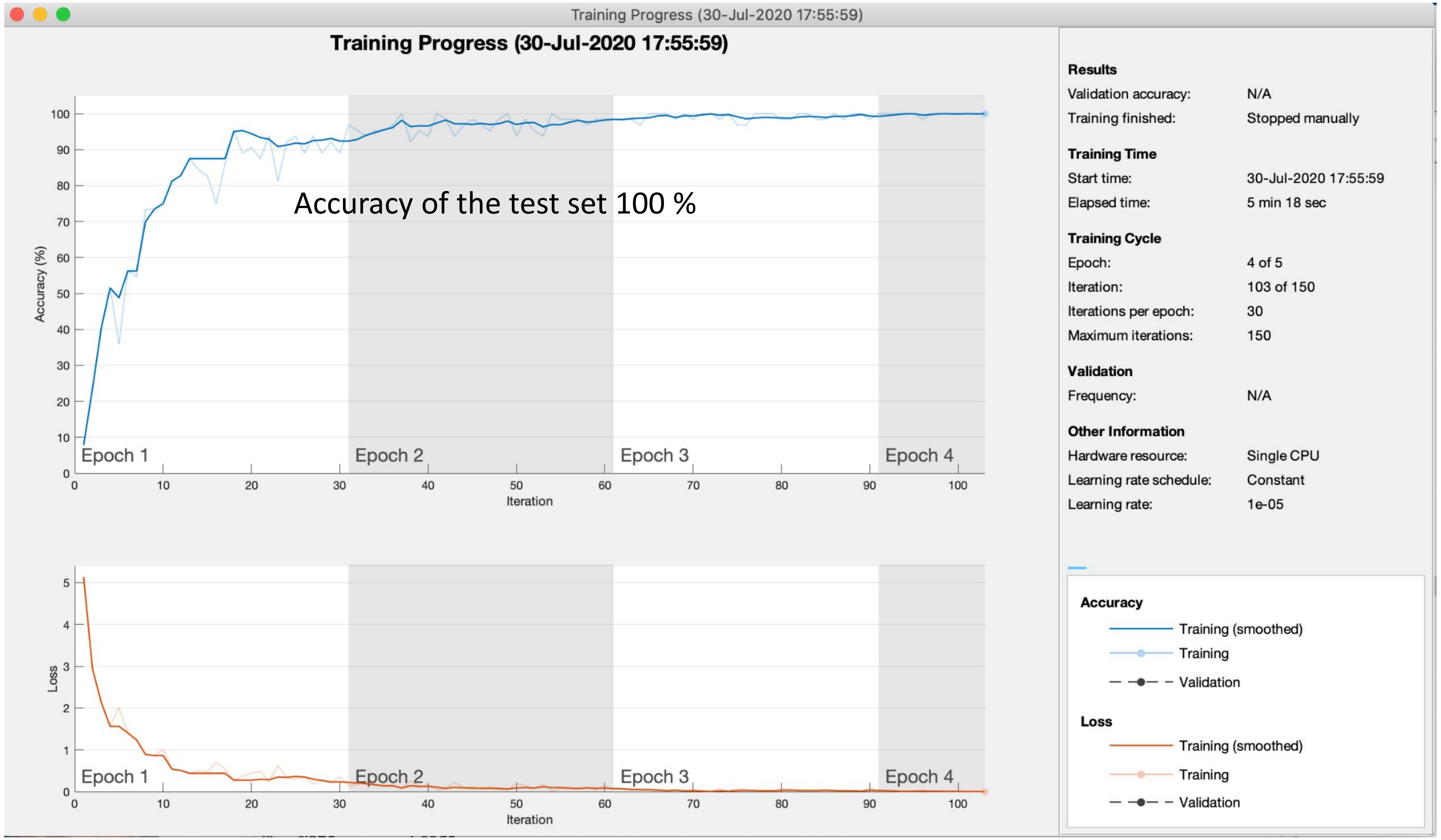
9 classes with 1956 images in the training set and 490 images in the test set (split 80%/20%).

Correct recognition, score 0.998

Person detected: Anatoly

Name	**Score**
{'Anatoly' }	0.99808
{'Cam' }	0.00025401
{'Charlie’ }	2.1408e-05
{'Eddie' }	4.8151e-05
{'Ernest' }	0.00054036
{'Michelle’}	0.0002557
{'Olga' }	5.351e-05
{'Vlad' }	0.00070837
{'Yelena' }	4.0724e-05

AlexNet CNN Training and Testing



Accuracy of the test set 100 %

Results

Validation accuracy:	N/A
Training finished:	Stopped manually

Training Time

Start time:	30-Jul-2020 17:55:59
Elapsed time:	5 min 18 sec

Training Cycle

Epoch:	4 of 5
Iteration:	103 of 150
Iterations per epoch:	30
Maximum iterations:	150

Validation

Frequency:	N/A

Other Information

Hardware resource:	Single CPU
Learning rate schedule:	Constant
Learning rate:	1e-05

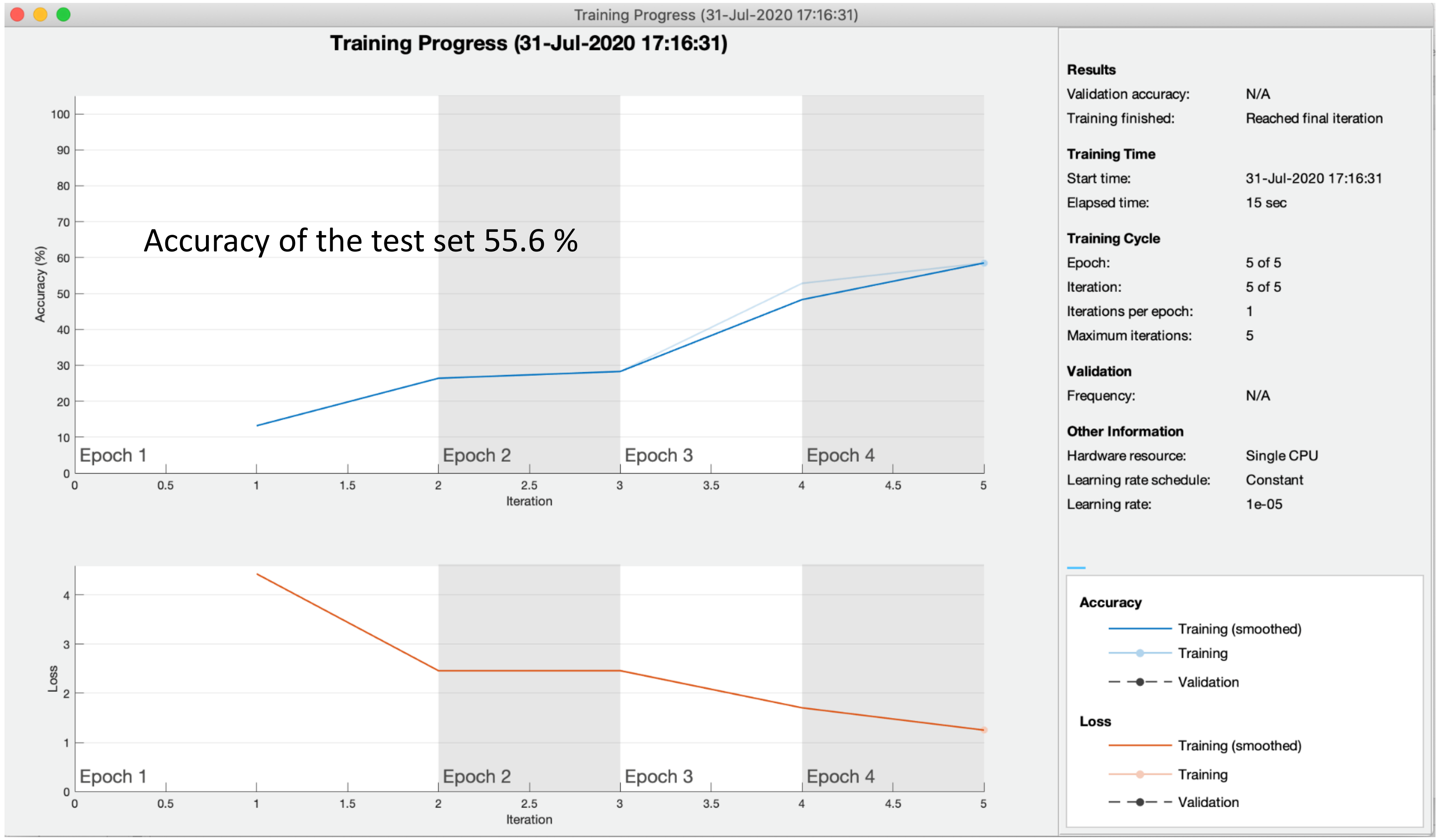
AlexNet CNN Face Detection and Recognition with a Reduced Image Database. Correct ID

9 classes with 7 images I each class.
Split 5/2 (training/test)
Correct recognition (score 0.56)

Person detected: Anatoly

Name	**Score**
{'Anatoly' }	0.55515
{'Cam' }	0.029337
{'Charlie' }	0.11275
{'Eddie' }	0.049929
{'Ernest' }	0.041123
{'Michelle'}	0.048524
{'Olga' }	0.012013
{'Vlad' }	0.059238
{'Yelena' }	0.091935

AlexNet CNN Training and Testing with a Small Number of Images (5 & 2). Correct ID



Results

Validation accuracy:	N/A
Training finished:	Reached final iteration

Training Time

Start time:	31-Jul-2020 17:16:31
Elapsed time:	15 sec

Training Cycle

Epoch:	5 of 5
Iteration:	5 of 5
Iterations per epoch:	1
Maximum iterations:	5

Validation

Frequency:	N/A

Other Information

Hardware resource:	Single CPU
Learning rate schedule:	Constant
Learning rate:	1e-05

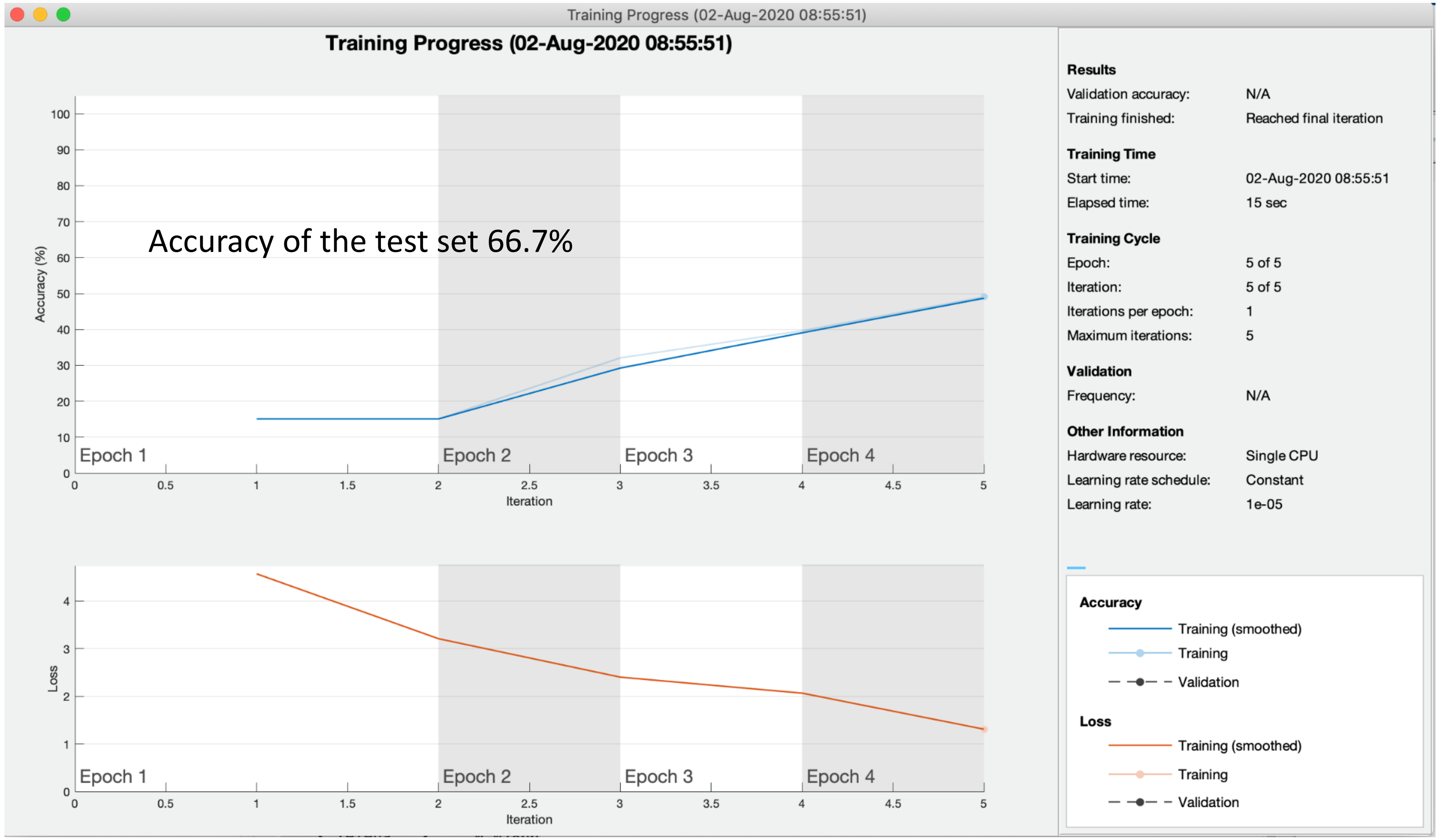
AlexNet CNN Face Detection and Recognition with a Reduced Image Database. Wrong ID

9 classes with 7 images I each class.
Split 5/2 (training/test)
Wrong recognition (score 0.484)

Person detected: Charlie

Name	**Score**
{'Anatoly' }	0.019589
{'Cam' }	0.033411
{'Charlie' }	0.48386
{'Eddie' }	0.03658
{'Ernest' }	0.039305
{'Michelle'}	0.20128
{'Olga' }	0.12124
{'Vlad' }	0.036063
{'Yelena' }	0.02866

AlexNet CNN Training and Testing with a Small Number of Images (5 & 2). Wrong ID



Accuracy of the test set 66.7%

Results

Validation accuracy:	N/A
Training finished:	Reached final iteration

Training Time

Start time:	02-Aug-2020 08:55:51
Elapsed time:	15 sec

Training Cycle

Epoch:	5 of 5
Iteration:	5 of 5
Iterations per epoch:	1
Maximum iterations:	5

Validation

Frequency:	N/A

Other Information

Hardware resource:	Single CPU
Learning rate schedule:	Constant
Learning rate:	1e-05

DMG Smart Tracking Project Status

1. Smart Track PDT application opens a video file and performs the following operations:
   - **Reading**: read a video file with a calibrated camera;
   - **Detection**: detect multiple objects of interest (faces, upper bodies, moving people) in a video frame (Cases 1 and 2);
   - **Recognition**: recognize faces and/or upper bodies of the people in the scene using a trained classifier (Case 1), preform labeling of the recognized people;
   - **Localization**: measure the real-world coordinates X,Y, and Z of people in the scene and calculate distances between the people;
   - **Prediction**: predict the object locations in the next frame using Kalman filtering (Case 2).
   - **Data association**: use the predicted locations (Case 2) and ground truth (Case 1) to associate detections across frames to form *tracks*.
   - **Timing**: use Date and Time Arithmetic capabilities with timestamps, calculate elapse time (Case 1).
2. Face recognition algorithms tested so far:
   - Singular value decomposition (SVD) provided a correct correct recognition rate of 83.3% for a small database (18 face images, 16 classes);
   - 2-D principal component analysis (2D PCA) provided a correct correct recognition rate of 92.5% for a database of 400 face images (40 classes);
   - MathWorks Aggregated Features from Stacked Images (AFSI) provided a correct recognition rate close to 100% for 16 face images (2 classes) – Demo2;
   - Histogram of oriented gradients (HOG) features provided a correct recognition rate of 97.5% with a database of 400 images (40 classes) – Demo2;
   - As a part of the Face Recognition Testbed development, a Deep Learning CNN (AlexNet) was trained and tested with 9 classes having total 1956 face images in the training set and 490 images in the test set (split 80%/20%). The correct recognition rate was close to 100% for input video frames that are not in the training/testing databases.

Summary and Conclusion

1. Preliminary results of training, test and evaluation of two similar Deep Learning CNN models used for face recognition are as follows:
    - **Quality of the training set** (number of images and their proper selection) is crucial for achieving high accuracy of face recognition;
    - **Accuracy** of face recognition declines with increasing the number of classes and decreasing the number of images in the face database.
2. It seems unlikely that the tested and evaluated Deep Learning CNN apps can achieve a reasonable performance with >6000 classes/persons and 1-2 pictures per person.
3. We have to continue our search for an optimum solution to the face recognition problem.

Suggested Face Recognition Algorithm for DMG Smart Track

TASK 59: Optimize selected model to meet the story use case criteria

Background and Motivation

1. **Goal**: find a reliable solution for the following use case: Recognize faces for provided video (Anatoly & Olga) using the labeled dataset of 40 classes with max 2 images per class.
2. **Acceptance Criteria**: Correct recognition rate greater or equal 85% for the Demo video (Anatoly & Olga) using a classifier trained with a face image database having 40 classes with max 2 images per class.
3. Face Recognition fundamentals and required steps/prerequisites:
   - An optimum choice of image segmentation (defining regions of interest), face detection, person recognition/ID, and tracking is crucial for meeting the acceptance criteria. Face detection and tracking are essential integral parts of any face recognition algorithm.
   - Our current application uses face detection algorithms, such as Viola-Jones or Kanade-Lucas-Tomasi (KLT) tuned to detect only frontal facial features. We need to properly configure and modify the face detection algorithms to include other classification models such as 'UpperBody', 'ProfileFace', etc.

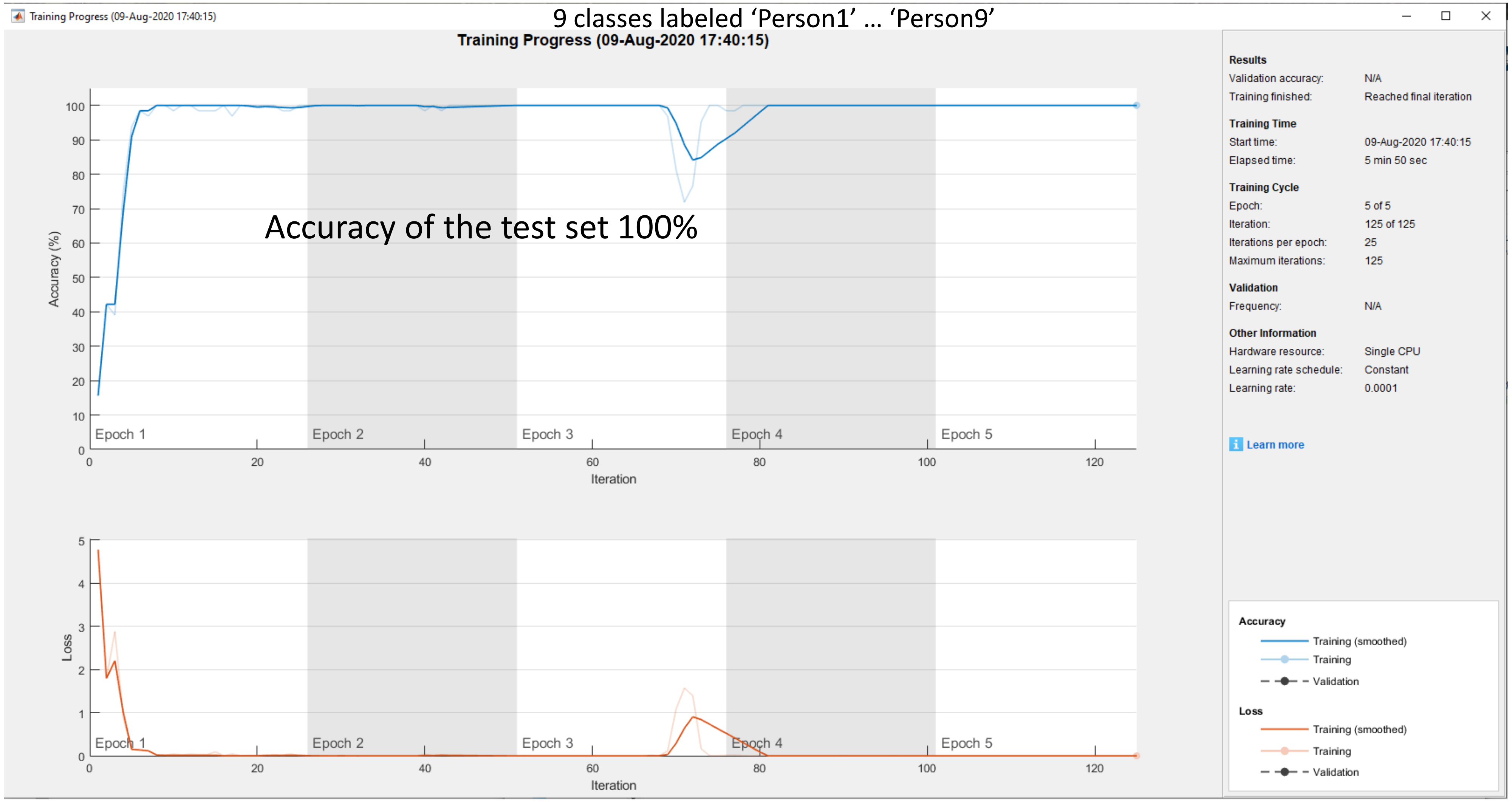
Algorithm Outline

1. Input data: max 2 images presumably taken right before boarding the cruise ship. A frontal face view and a profile view would be preferable.
2. Train and test the properly chosen face recognition algorithm on video surveillance data, e.g. collected during the boarding and assign temporary labels, e.g. ‘Person1’, ‘Person2’, … ‘PersonN’, to each passenger and crew member to generate a face image database.
3. Run face recognition with the collected 1 or 2 face (presumably 1 frontal view and 1 profile view) to assign correct label/ID to each person/class.

Training and Test Face Image Databases

1. Training database contains 9 classes of unnamed persons labeled as ‘Person1’ through ‘Person9’
   - 227 face images of ‘Person1’ acquired from the video 'Movie on 6-11-20 at 12.43 PM.mov’ (Anatoly);
   - 248 face images of ‘Person2’ acquired from the video 'Movie on 6-11-20 at 12.43 PM.mov’ (Olga);
   - 414 face images of ‘Person3’ acquired from the videos ‘Cam with glasses.MOV’ and ‘Cam without glasses.MOV’ (Cam);
   - 165 face images of ‘Person4’ acquired from the video ‘Movie on 7-7-20 at 2.07 PM.mov’ (Charlie);
   - 165 face images of ‘Person5’ acquired from the video ‘Movie on 7-8-20 at 11.05 AM.mov’ (Eddie);
   - 83 face images of ‘Person6’ acquired from the video ‘Movie on 7-9-20 at 11.53 AM.mov’ (Ernest);
   - 212 face images of ‘Person7’ acquired from the video ‘Michelle.MOV’ (Michelle);
   - 240 face images of ‘Person8’ acquired from the video ‘Vlad.MOV’ (Vlad);
   - 316 face images of ‘Person9’ acquired from the videos ‘Lena1.MOV’ and ‘Lena2.MOV’ (Yelena);
2. Face recognition was tested with a test set containing 2 images of Anatoly and 2 images of Olga that were not in the training database.

Training AlexNet with Video Surveillance Data


Training Progress (09-Aug-2020 17:40:15)
9 classes labeled 'Person1' … 'Person9'
Training Progress (09-Aug-2020 17:40:15)
Accuracy of the test set 100%
Accuracy (%)
Loss
Iteration
Epoch 1
Epoch 2
Epoch 3
Epoch 4
Epoch 5

Results

Validation accuracy:	N/A
Training finished:	Reached final iteration

Training Time

Start time:	09-Aug-2020 17:40:15
Elapsed time:	5 min 50 sec

Training Cycle

Epoch:	5 of 5
Iteration:	125 of 125
Iterations per epoch:	25
Maximum iterations:	125

Validation

Frequency:	N/A

Other Information

Hardware resource:	Single CPU
Learning rate schedule:	Constant
Learning rate:	0.0001

Learn more

Accuracy
- Training (smoothed)
- Training
- Validation

Loss
- Training (smoothed)
- Training
- Validation

Correctly Recognizing ‘Person1’ as Anatoly

- **Person detected: ‘Person1’**
- *Recognized as ‘Anatoly’*

Name	Score
{'Person1’}	1
{'Person2’}	1.2651e-08
{'Person3’}	6.1211e-08
{'Person4’}	2.2432e-09
{'Person5’}	2.2946e-09
{'Person6’}	9.7534e-08
{'Person7’}	1.4534e-08
{'Person8’}	2.2378e-08
{'Person9’}	2.1006e-08

Correctly Recognizing ‘Person2’ as Olga

- **Person detected: ‘Person2’**
- *Recognized as ‘Olga’*

Name	Score
{'Person1’}	2.424e-07
{'Person2’}	0.99999
{'Person3’}	8.851e-07
{'Person4’}	1.4541e-07
{'Person5’}	7.509e-06
{'Person6’}	3.1566e-07
{'Person7’}	2.556e-08
{'Person8’}	6.4212e-08
{'Person9’}	3.8412e-07

Summary and Conclusion

1. Preliminary results of training, test and evaluation of the DMG Smart Track face recognition approach using a Deep Learning CNN classifier trained with face images extracted from surveillance videos are as follows:
   - **A set of face images** acquired during detection and tracking of multiple people can be used to train a classifier to achieve a correct recognition rate of 100% for 9 subject classes;
   - This approach may achieve a reasonable performance with >6000 classes/persons and 1-2 pictures per person.
2. Next step will be adding more classes to the training database, presumably, using surveillance videos available elsewhere.