Sports Camera Calibration via Synthetic Data: https://github.com/lood339/SCCvSD
Pytorch-Two-Gan: https://github.com/lood339/pytorch-two-GAN

How to setup:

Step 1: Open a terminal in the SeniorDesign folder
Step 2: conda env create -f environment.yml
Step 3: conda activate sd
Step 4: conda install hdf5storage
Step 5: If using local database: change run.py line 252 to ‘localhost’
Step 6: python run.py --game 0 --video './segmentation/datasets/video2.mp4’

Overview of all files with parameters:

Run.py:

- Used to run the entire algorithm from start to finish.
- Loads the 3 models
    - Segmentation model
    - Feature model
    - Homography model
- Command line arguments:
    - --game
        - Game_id number
    - --video
        - Path to video
- Parameters:
    - Step_value
        - Generate a homography every step_value. This is used so we don’t need to generate a homography every single frame but rather every x number of frames. The intuition behind this is that within a fraction of a second, a person in real life is not able to move a considerable amount and thus we can just infer where the player was during that time
        - Currently uses a value of 5 to generate a homography every 5 frames
        - If using a larger fps than 15, increase step_value subsequently

segmentation/Segmentation_model.py:

- Loads the segmentation model into memory

feature_generation/Generate_features.py:

- Loads the feature model into memory
- Gen_features function
    - returns the generated feature of the frame

Homography.py:

- This is used to generate homographies and coordinates of the input
- Generate_homography function
    - generates a homography by querying the database of 100k images
- Generate_coordinates function
    - Warps all the coordinates in the frame using the homography
- Interpolate_coordinates function
    - Interpolates player coordinates between the first homography and second homography

deep/Custom_database.py

- Used to generate a custom database using given parameters
- Parameters:
    - Volleyball.mat
        - Mat file that has field intersections and points
    - Worldcup_dataset_camera_parameters.mat
        - Mat file that has camera parameters
    - Camera parameter values:
        - Cc_mean
            - mean location distribution of the camera center
        - Cc_std
            - std location distribution of the camera center
        - Cc_min
            - min location distribution of the camera center
        - Cc_max
            - max location distribution of the camera center
        - Fl_mean
            - mean of the focal length distribution
        - Fl_std
            - std of the focal length distribution
        - Fl_min
            - min of the focal length distribution
        - Fl_max
            - max of the focal length distribution
        - Roll_statistics
            - base_roll: camera base, roll angle
        - Pan_range
            - range of camera panning
        - Tilt_range
            - range of camera tilt
    - Num_camera:
        - Number of generated synthetic images wanted
        - 91173 will load into memory about 30GB worth of images
- Pivot_cameras
    - Generated cameras using the camera parameters
- Positive_cameras
    - Pivot_cameras that are slightly randomized

deep/network_test.py

- Used to generate features for a new database
- Parameters
    - Save_file
        - Name and path of where to save the features to

Running the algorithm:

This file is used to run the entire algorithm from start to finish provided the data is there already.
This is run simply via the command: python run.py --game 0 --video ‘path/to/video.mp4’
What it does:

Step 1: Connect to the database
Step 2: Load the coordinate information into memory
Step 3: Load the segmentation, feature, and homography models
Step 4: While loop:

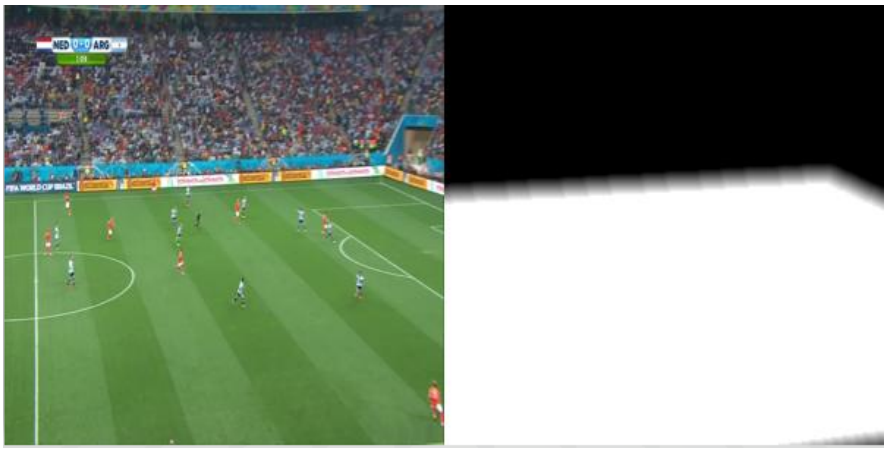
- Do the necessary image transformations
- Set the input into the segmentation model
- Return the segmented image
- Generate a feature of the segmented image
- Generate a Homography of the image using the feature
    - If increment is at step_value: interpolate coordinates
    - Insert values into the database
- Repeat Step 4

Training the segmentation model:

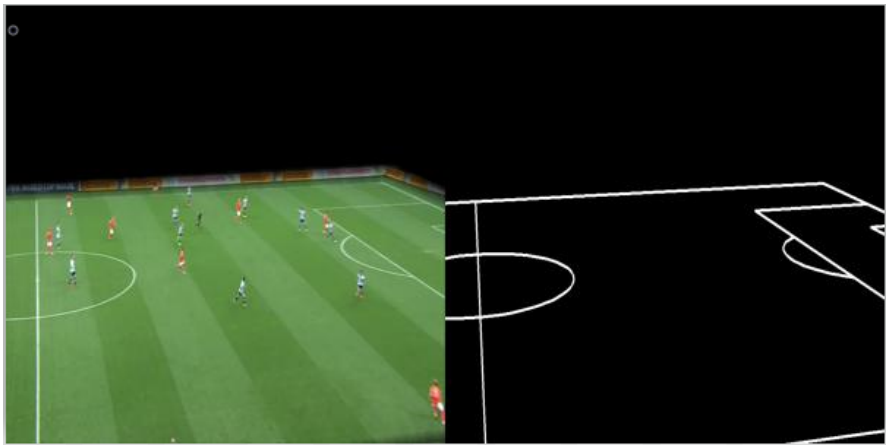
In order to train the segmentation model on new data, you will need to annotate two sets of images: train_phase_1, and train_phase_2.

Step 1:

Train_phase_1 consists of a side-by-side image of the frame and field segmented out.
Name the image 001_AB, 002_AB, etc.. under SeniorDesign/segmentation/datasets/soccer_seg_detection/train_phase_1/



Train_phase_2 consists of a side-by-side image of the frame and the field lines segmented out:
Name the image 001_AB, 002_AB, etc.. under SeniorDesign/segmentation/datasets/soccer_seg_detection/train_phase_2/



Step 2:

In a terminal under SeniorDesign/segmentation, edit training.sh arguments to match your directories.
--dataroot: path to the train_phase_1 and train_phase_2
--checkpoints_dir: path to save the checkpoints to
--niter: number of training iterations

```
python train_two_pix2pix.py \
--dataroot ./datasets/soccer_seg_detection \
--checkpoints_dir /home/brendan/extra/checkpoints \
--name soccer_seg_detection_pix2pix \
--model two_pix2pix \
--which_model_netG unet_256 \
--which_direction AtoB \
--lambda_A 100 \
--dataset_mode two_aligned \
--no_lsgan \
--norm batch \
--pool_size 0 \
--output_nc 1 \
--phase1 train_phase_1 \
--phase2 train_phase_2 \
--save_epoch_freq 10 \
--niter 500
# --continue_train True
```

Run the bash script training.sh via: bash training.sh

Expanding to other sports:

1. Train the segmentation model with the annotated images

2. Create a mat file that has the line segments and intersections and put it into SeniorDesign/data/.
    a. Use SeniorDesign/data/worldcup2014.mat as an example

line_segment_index	uint16	151x2	[0, 1; 2, 3; 4, 5; 6, 7; 8, 9
points	double	302x2	[0, 0; 26.250, 0; 26.250, 0; 52.500, 0; 52.500, 0

    b. Line_segment_index
        - Pairings of which values in points correspond to a line
        - Line_segment_index[0] = [0, 1]
            - Corresponds to points 0 and 1 forming a line

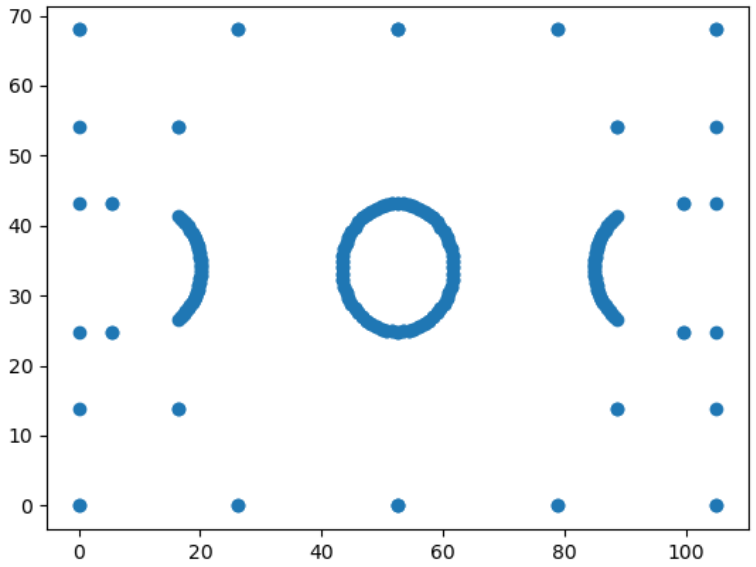
line_segment_index

	1	2
1	0	1
2	2	3
3	4	5
4	6	7
5	8	9
6	10	11

        - Line_segment_index will always be exactly half of points since you need two points to form a line between
    c. Points
        - X, y values of the points on the field
        - Points[0] = [0, 0], Points[1] = [26.25, 0]
            - There is a point at (0,0) and at (26.25, 0)
            - This will form a line between the two points

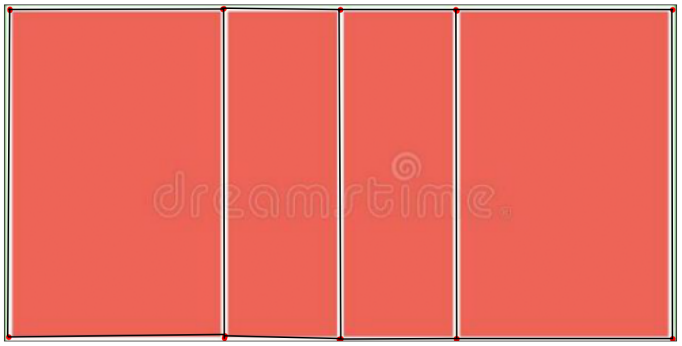
-

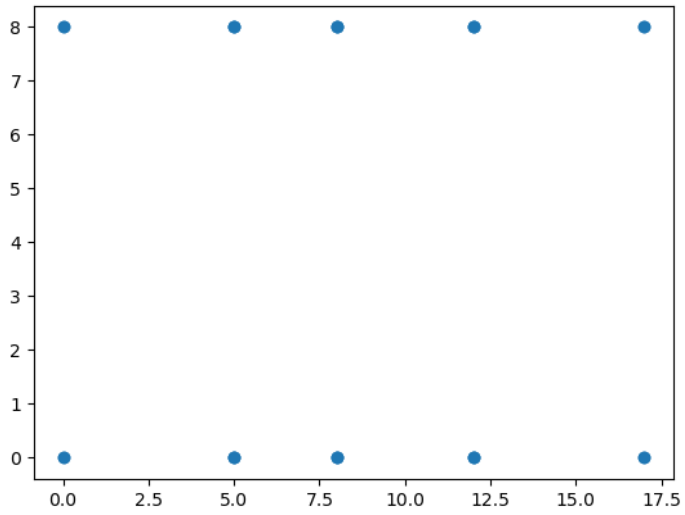
points		
	1	2
1	0	0
2	26.25	0
3	26.25	0
4	52.5	0
5	52.5	0
6	78.75	0
7	78.75	0
8	105	0
9	0	68.004
10	26.25	68.004

- Points must be in range:
  - X: 0 to length of field
  - Y: 0 to height of field
  - E.g. Soccer field is 115x74 so these points are between those values
- If you were to plot all the points, you will get this which will represent the field
- 

- Put this in SeniorDesign/data

d. I created a tool under SeniorDesign/Calibration/field_points.py which is an easy way to do this step:
    - If using soccer/volleyball/football, change line 13 to whichever one. Otherwise, include your own that has a path to the top-down view of the field, the height of the field in meters, and the width in meters
    - On line 86, change ‘data/football/mat’ to whatever you want the matfile to be called
    - Make the points as exact as you can for better accuracy





3. Change parameters in SeniorDesign/deep/custom_database.py
    a. Change line 60 to be the location of the field map you created in the previous step

b. Change line 61 to whatever you named your points to be in the mat e.g. ‘points’, ‘court_points’, ‘line_points’.
c. Change line 62 to whatever you named the line segments to be in the mat
d. If you have a mat file that already has camera parameters, change line 67 to that file. Otherwise, change all the values of the camera parameters to the values you determined worked with your field.
    - Lines 100-103 are manually entered values that work with volleyball
    - If you want to run it with soccer, remove 100-103
    - If you want to run it with your own data, change 100-103
    - I am unsure of what roll_statistics do so I have left them at their previous values although I believe they are rotation vectors
e. Change num_camera at line 115 to however many synthetic images you want
f. Change the name of the save file at line 179 to whatever you want it to be called
g. Run python custom_database.py
    - This will generate synthetic images of the field using the camera parameter statistics and the field map from step 2

4. Change generate_database_features.sh parameters
    a. Edge-image-file is the matfile created in step 3
    b. Save-file is the location to save the feature database to
    c. In a terminal, run: bash generate_database_features.sh
5. Change names in SeniorDesign/homography.py
    a. Line 15: Mat file created in step 3
    b. Line 20: Mat file created in step 2
6. Run python run.py --game 0 --video './segmentation/datasets/video2.mp4'