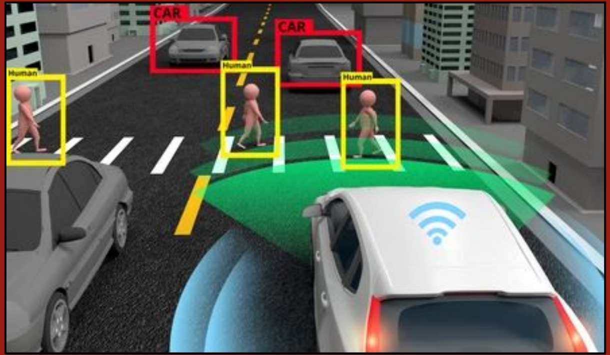
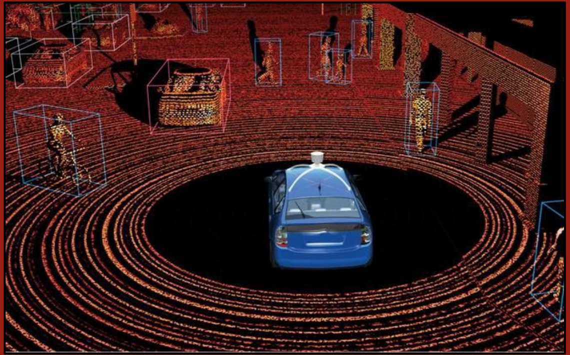
Pedestrian Detection and Sensor Fusion





Luke Davidson, Arvinder Singh, Uday Raghuvanshi, Lucas Eby

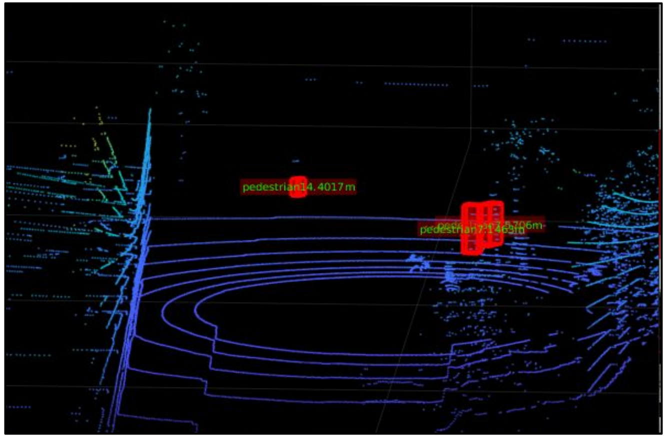
Northeastern University

Overview

1. Detection and Fusion Goal
2. Technology
3. Data Collection
4. Detection: YOLOv5
5. Sensor Fusion
    a. LiDAR and RGB
    b. LiDAR and IR
    c. RGB and IR
6. Limitations/ Challenges
7. Questions

Detection + Fusion Goal

- **Problem:** All sensors have individual strengths and weaknesses



- **Goal:** Combine the strengths of different sensors to increase the efficiency and accuracy of detecting objects at any moment in any environment

Luke Davidson, Arvinder Singh, Uday Raghuvanshi, Lucas Eby

Northeastern University

Technology

Velodyne VLP 16 LiDAR

- Extremely accurate distance and position measurements

Ouster OS1 LiDAR

- Lots of noise
- Not reliable for object detection
- Higher cost: computation and price

Teledyne FLIR ADK IR Camera

- Can often get a picture when RGB can't
- Lower resolution image
- Detection not as efficient

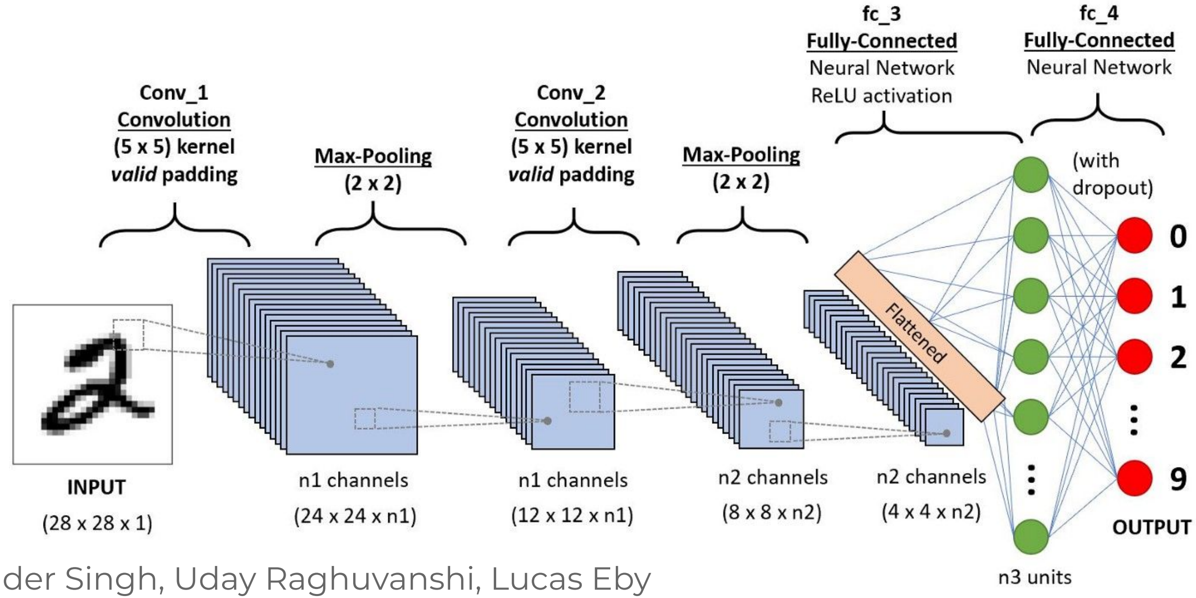
Teledyne FLIR BlackflyS 5MP Camera

- High resolution
- Very effective in right environment
- Low cost
- Performs poorly in non-ideal conditions

- Software
  - MATLAB
  - Python

Convolutional Neural Networks(CNN)

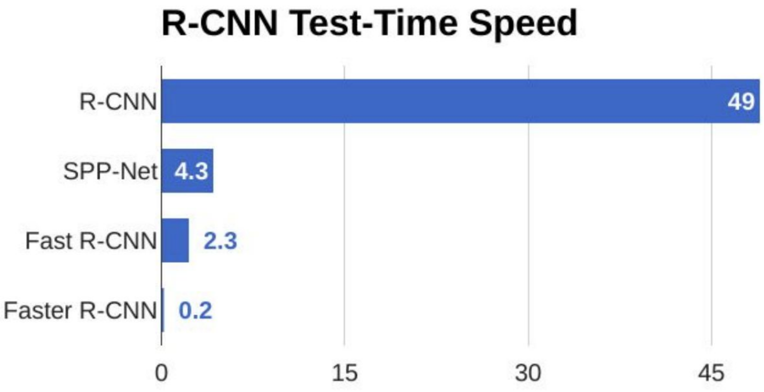
- Cameras are relatively cheap and have high resolution
- Layers are made up of filters that can detect "patterns"
- BUT the size of the object is not guaranteed to be constant

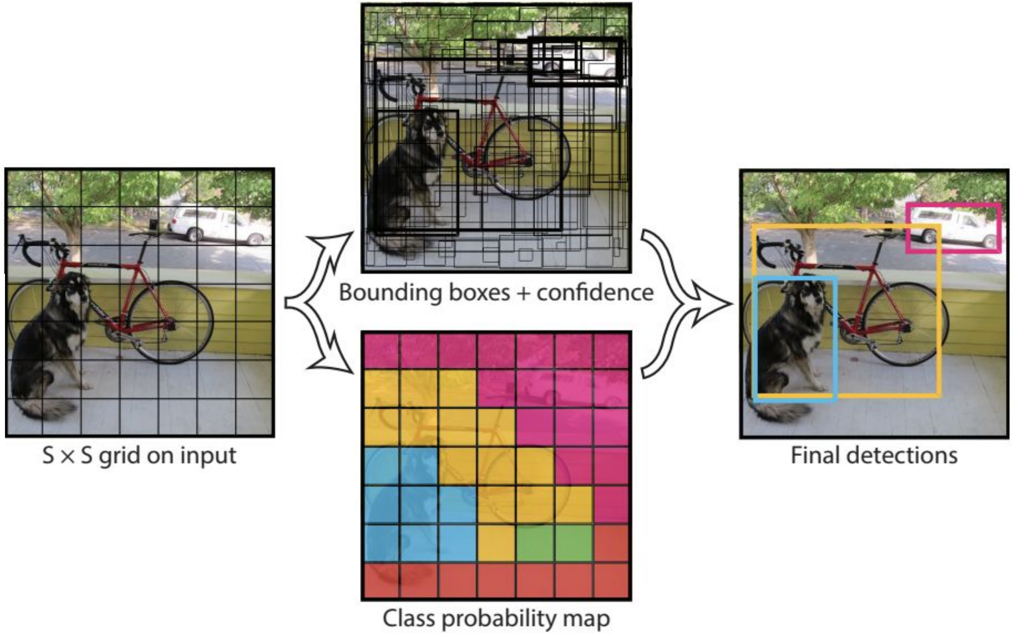


Luke Davidson, Arvinder Singh, Uday Raghuvanshi, Lucas Eby

You Only Look Once (YOLO)

- A CNN that only uses a single propagation through a neural network to detect objects
- Made up of:
  1. Residual blocks
  2. Bounding box regression
  3. Intersection over union





Luke Davidson, Arvinder Singh, Uday Raghuvanshi, Lucas Eby

Northeastern University

IR vs RGB

Collected Night Data

Provided Day Data

Luke Davidson, Arvinder Singh, Uday Raghuvanshi, Lucas Eby

Northeastern University

Data Collection Checklist

- Make sure the required sensors are clean.
- Check whether all the ros drivers are running.
- Check whether your rosbag record command has all the topics you need.
- Make sure that the LiDARs are turned on by checking the display console in the front center of car.
- Rostopic echo all the topics once you start recording to ensure you have incoming data and visualize in parallel with Rviz.
- Make sure the car's hdd has enough space to store your rosbags.
- Before you drive, make sure the car has enough fuel to run
- Be patient while using the keyboard (very important).
- Try your best to not crash the car (most important)

Luke Davidson, Arvinder Singh, Uday Raghuvanshi, Lucas Eby

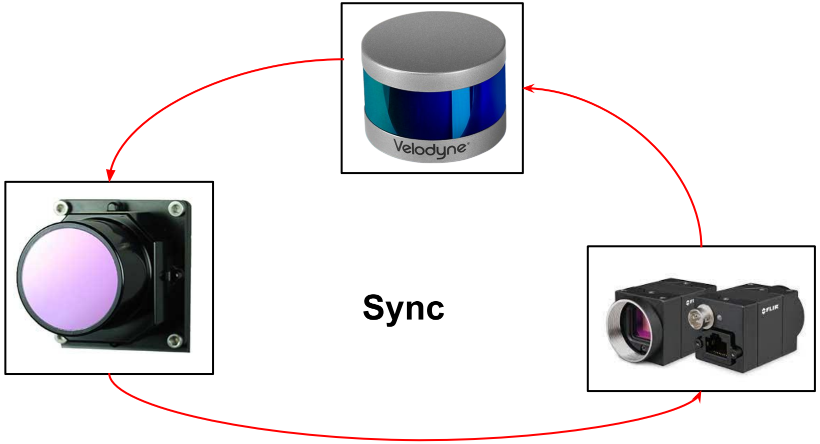
Northeastern University

Sensor Fusion

1. **Time Syncing**

   Output rate of the sensors:

   - RGB Camera: 8Hz
   - IR Camera: 30Hz
   - Velodyne VLP16 LiDAR: 10Hz
   - Ouster OS1 LiDAR: 10Hz

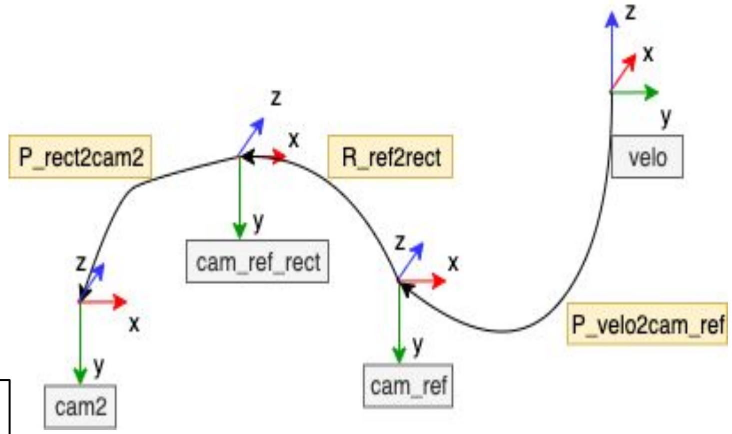



```
camera = select(bag,'Topic','/camera_array/cam0/image_raw');
lidar = select(bag,'Topic','/ns1/velodyne_points');
ir = select(bag,'Topic','/boson_camera_array/cam_right/image_raw');


cameraMsgs = readMessages(camera);
lidarMsgs = readMessages(lidar);
irMsgs = readMessages(ir);

ts1 = timeseries(camera);
ts2 = timeseries(lidar);
ts3 = timeseries(ir);

t1 = ts1.Time;
t2 = ts2.Time;
t3 = ts3.Time;

k = 1;
k2 = 1;

if size(t2,1) > size(t1,1)
    for i = 1:size(t1,1)
        [val,indx] = min(abs(t1(i) - t2));
        if val <= 0.1
            idx(k,:) = [i indx];
            k = k + 1;
        end
    end
else
    for i = 1:size(t2,1)
        [val,indx] = min(abs(t2(i) - t1));
        if val <= 0.1
            idx(k,:) = [indx i];
            k = k + 1;
        end
```

Luke Davidson, Arvinder Singh, Uday Raghuvanshi, Lucas Eby

2. Transformations

- Lidar to Camera
  1. Find Camera intrinsics (yaml)
  2. Get available frames from ROS
  3. Calculate the transformation matrix using getTransform
- Camera to Lidar- invert(LidarToCamera)




```
%Transformations
frames = bag.AvailableFrames;
gettf= getTransform(bag,frames{21},frames{4});% cam_0_optical_frame and vlp_16 port (frame 18,os_sensor for Ouster)
tf = gettf.Transform;
ros_quat = tf.Rotation;
quat = [ros_quat.X ros_quat.Y ros_quat.Z ros_quat.W];
rotm = quat2rotm(quat);

ros_trans = tf.Translation;
trans_RGB = [ros_trans.X ros_trans.Y ros_trans.Z];
trans_IR = [-0.8 -0.23 0];
tform = rigid3d(rotm, trans_RGB); %RGB transformation matrix
tform=rigid3d(rotm,trans_IR);% IR transformation matrix
```

Finding transformation matrices

```
% IR Intrinsics
focalLength_IR = 2*[5.24888150200348832e+02, 5.21776791343664968e+02];
principalPoint_IR = 2*[3.25596989785447420e+02, 3.25596989785447420e+02];
%RGB Intrinsics
focalLength_RGB = [1888.4451558202136, 1888.4000949073984];
principalPoint_RGB = [613.1897651359767, 482.1189409211585];
```

Camera Intrinsics

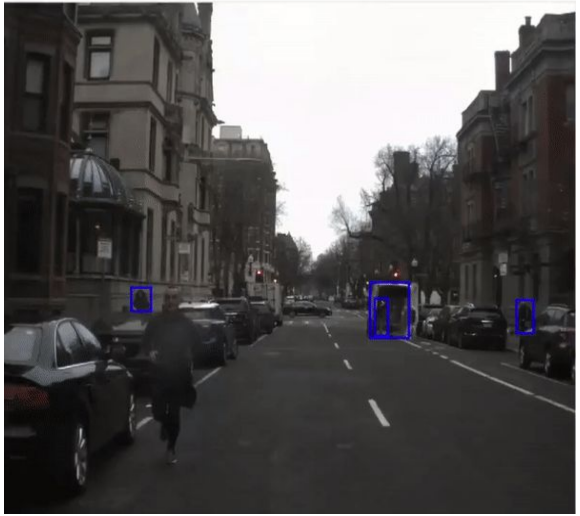
Luke Davidson, Arvinder Singh, Uday Raghuvanshi, Lucas Eby

Northeastern University

3. Projecting bounding boxes on RGB/IR Image

- Extract bounding boxes location from YOLO
- Pre-process the data to get only person class
- Normalize YOLO coordinates
- Convert relative coordinates to pixel coordinates

```
fileID = fopen(labels_loc,'r');
formatSpec = '%f';
A = fscanf(fileID,formatSpec);
k = 1;
clusterpoints = [];
if length(A)>=5
    for i = 1:5:length(A)
        if A(i)==0
            x_center= A(i+1,1)*w;
            y_center= A(i+2,1)*h;
            width= A(i+3,1)*w;
            height= A(i+4,1)*h;
            xLeft = x_center - width/2;
            yLeft = y_center - width/2;
            xBottom = x_center - height/2;
            yBottom = y_center - height/2;
            bbox = [xLeft, yBottom, width, height];
            figure(1)
            hold on
```



Code **Results**

Luke Davidson, Arvinder Singh, Uday Raghuvanshi, Lucas Eby

4. Segment 3D point cloud data on 2D RGB/IR image

- Downsample ? No !
- Pre-Process ? Yes !

➔ Extract lidar points inside the 2D bbox

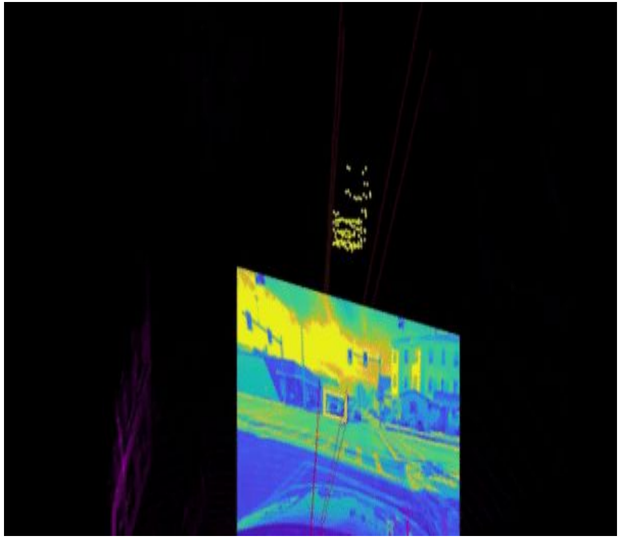
Pre-Process

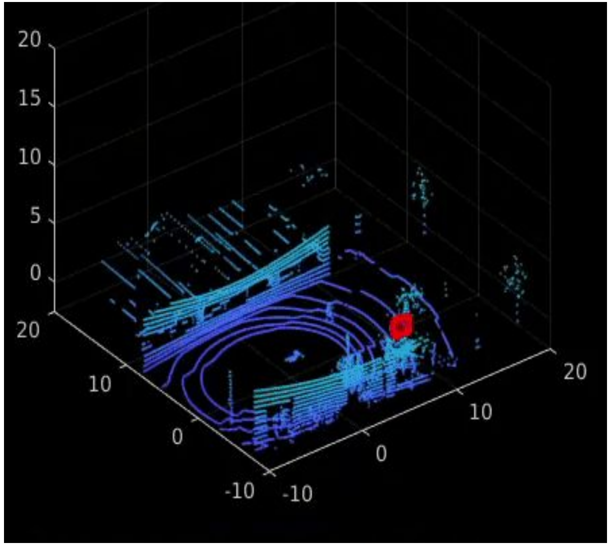
```
for j = 1:length(imPts)
    if imPts(j,1)<=xLeft+width &&  imPts(j,1)>=xLeft
        if imPts(j,2)<=yBottom+height &&  imPts(j,2)>=yBottom
            clusterpoints(k, 1) = imPts(j,1);
            clusterpoints(k, 2) = imPts(j,2);
            k = k+1;
        end
    end
end
```

Result

5. Adding 3D bounding boxes in point cloud data

- Find transformation from camera frame to lidar frame [invert(LidarToCamera)]
- Define Cluster Threshold
- Transform corners of 2D bboxes to 3D lines (Frustum Flaring)
- Segment into various clusters based on Euclidean distance





Luke Davidson, Arvinder Singh, Uday Raghuvanshi, Lucas Eby

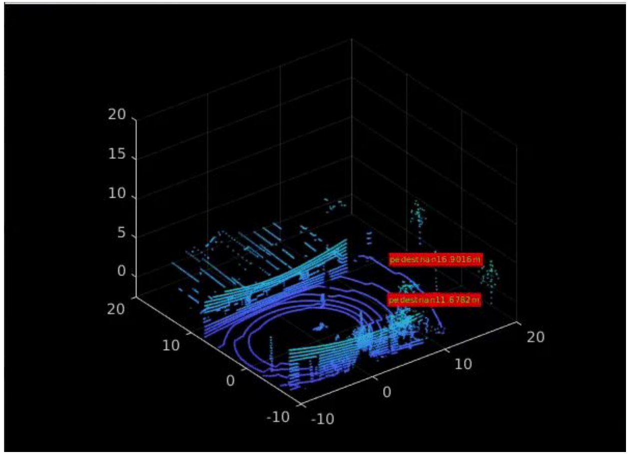
Northeastern University

6. Calculating distance of pedestrians from car

- Extracting the coordinates of 3D bounding boxes.
- Keeping only distances of interest
- Calculating distance from LiDAR
- Using the distances in autonomous driving

Code

```
bboxLidar = bboxCameraToLidar(bbox,pc,intrinsics,invert(tform),'ClusterThreshold',1);
bbox=[floor(xLeft), floor(yBottom), floor(width), floor(height)];
dist_x=bboxLidar(:,1);
dist_y=bboxLidar(:,2);
dist_z=bboxLidar(:,3);
if isempty(dist_x)==false
    dist=sqrt(dist_x^2+dist_y^2+dist_z^2)-2;
    labels="pedestrian"+dist+"m";
    figure(1)
    hold on
    showShape('rectangle',bbox,'color','yellow','Label',labels)

    figure(2)
    hold on
    showShape('cuboid',bboxLidar,'Opacity',0.5,'Color','red', ...
        'Label',labels,'LabelTextColor','green','LabelFontSize',7,'LabelOpacity',0.3)

end
```

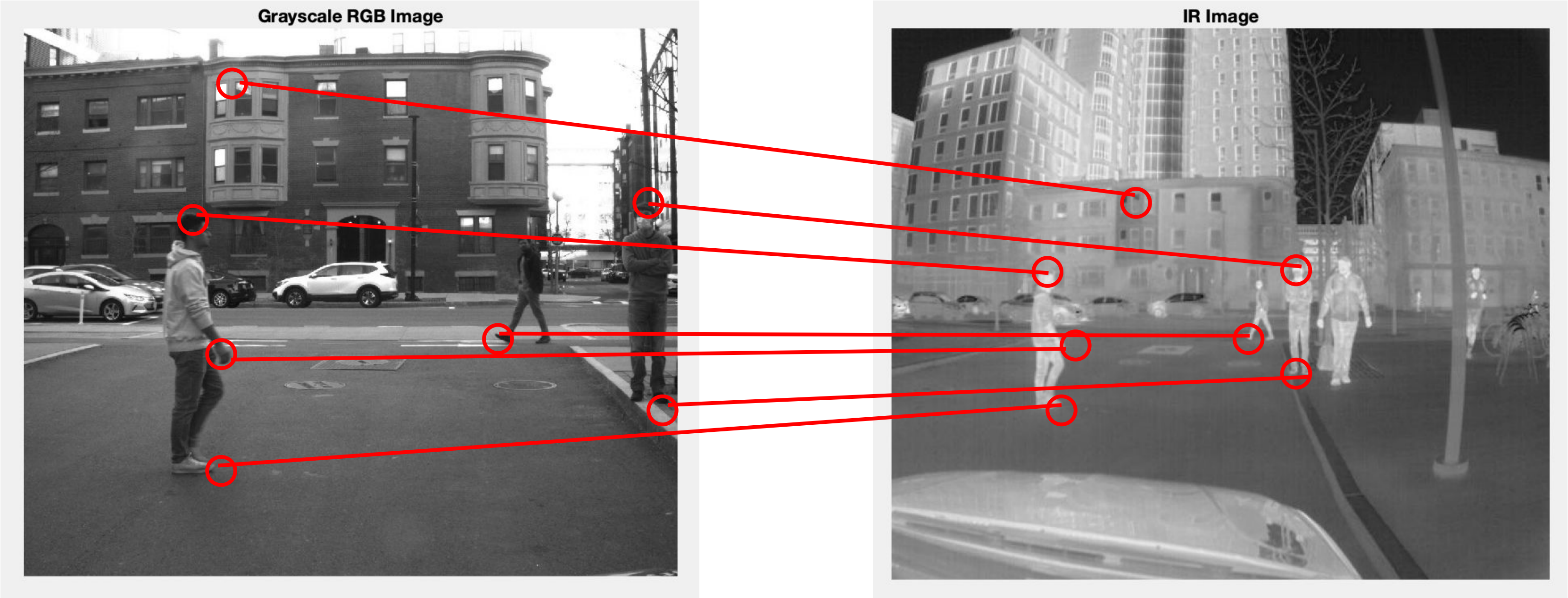


Luke Davidson, Arvinder Singh, Uday Raghuvanshi, Lucas Eby

Northeastern University

Sensor Fusion

Fusing RGB and IR



Luke Davidson, Arvinder Singh, Uday Raghuvanshi, Lucas Eby

Northeastern University

Sensor Fusion

Fusing RGB and IR

Read and Convert

```
ir_im = imread('/Users/lukedavidson/Desktop/ir_im_box2.png');
rgb_im = imread('/Users/lukedavidson/Desktop/rgb_im_box2.png');
rgb_to_gray = rgb2gray(rgb_im);
```

Resize

```
size_rgb = size(rgb_to_gray);
ir_im_resize = imresize(ir_im,size_rgb);
```

Create TF

```
movingPoints = [765 853; 593 889; 699 647; 661 389; 527 579; 605 697];
fixedPoints = [499 735; 399 743; 461 619; 447 479; 381 577; 419 643];
tform = fitgeotrans(movingPoints,fixedPoints,'projective');
```

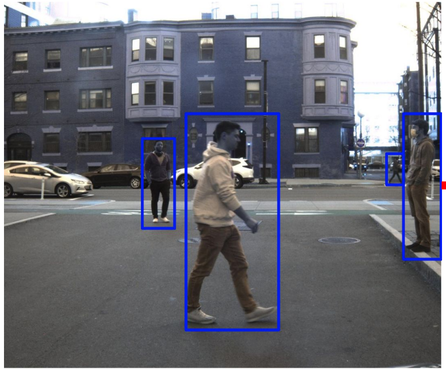
Transform and Display

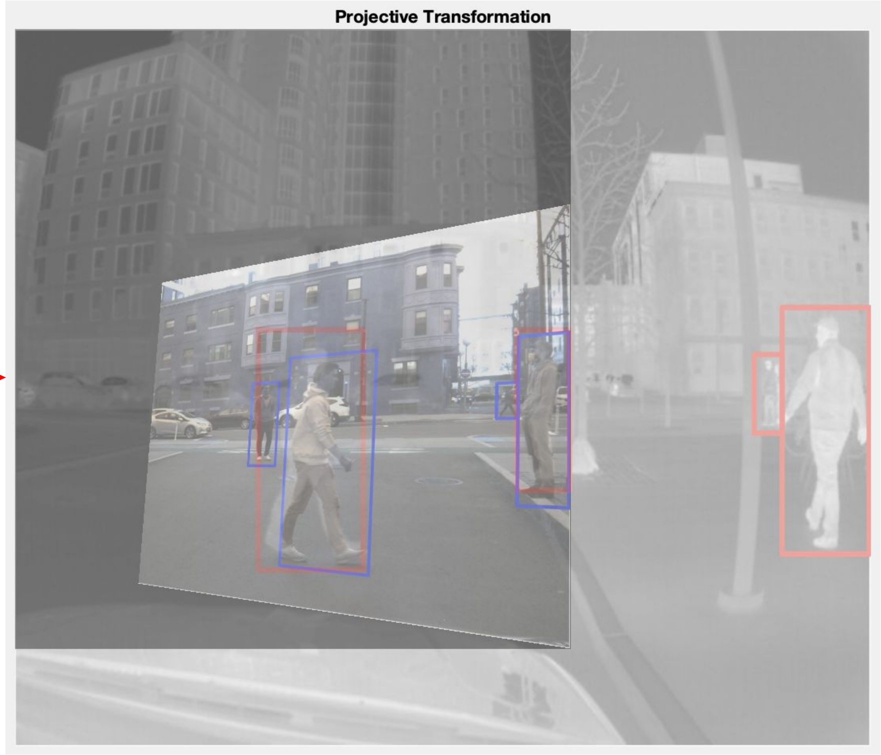
```
rgb_warped = imwarp(rgb_im,tform);
rgb_warped_translated = imtranslate(rgb_warped,[155,220],"OutputView","full");
figure
imshow(ir_im_resize)
alpha(0.4)
hold on
imshow(rgb_warped_translated)
alpha(0.4)
title('Projective Transformation')
```

Luke Davidson, Arvinder Singh, Uday Raghuvanshi, Lucas Eby

Northeastern University

Sensor Fusion





Luke Davidson, Arvinder Singh, Uday Raghuvanshi, Lucas Eby

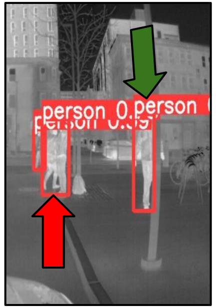
Northeastern University

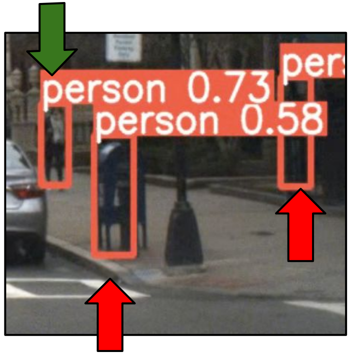
Limitations/Challenges

1. Velodyne vs Ouster- Having better LiDAR increases cost.
2. YOLO is not perfect
   - False positives
   - False Negatives
   - Single box for multiple pedestrians
3. Parameters required for calibration
   - Data set had older calibration values

Ouster

Velodyne

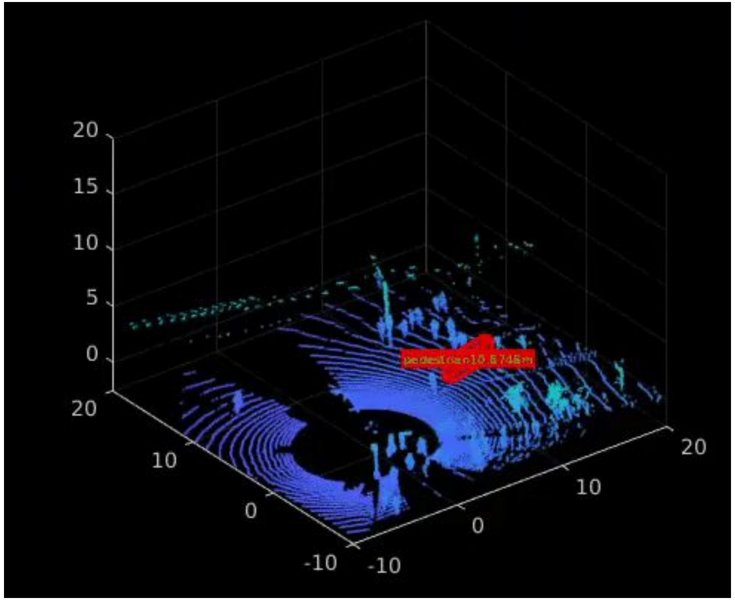




Luke Davidson, Arvinder Singh, Uday Raghuvanshi, Lucas Eby

Northeastern University

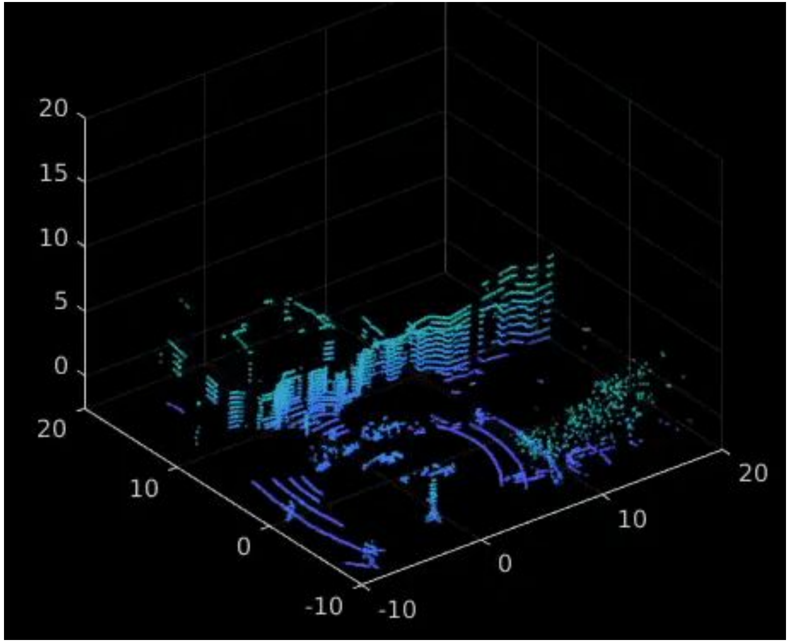
Final Outcome (RGB)



Luke Davidson, Arvinder Singh, Uday Raghuvanshi, Lucas Eby

Northeastern University

Final Outcome (IR)



Luke Davidson, Arvinder Singh, Uday Raghuvanshi, Lucas Eby

Conclusion

- We were able to get a rough transformation between IR and Velodyne Lidar
- We were able to detect the poses of pedestrians up until about 15 meters with high precision
- We had about 85-90% accuracy with YOLO
- If we had more time in the future we would:
  - Calibrate the velodyne and ouster as well as the RGB and IR cameras together
  - Add path prediction to each pedestrian

Questions?

Works Cited

Bandyopadhyay, Hmrishav. "Yolo: Real-Time Object Detection Explained." *V7Labs*, 19 Mar. 2022, https://www.v7labs.com/blog/yolo-object-detection.

DeepAI. "Max Pooling." *DeepAI*, 17 May 2019, https://deepai.org/machine-learning-glossary-and-terms/max-pooling.

Gandhi, Rohith. "R-CNN, Fast R-CNN, Faster R-CNN, YOLO - Object Detection Algorithms." *Towards Data Science*, 9 July 2018, https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e.

Ganesh, Prakhar. "Types of Convolution Kernels Simplified." *Towards Data Science*, 18 Oct. 2019, https://towardsdatascience.com/types-of-convolution-kernels-simplified-f040cb307c37.

Karimi, Grace. "Introduction to Yolo Algorithm for Object Detection." *Section*, 15 Apr. 2021, https://www.section.io/engineering-education/introduction-to-yolo-algorithm-for-object-detection/.

Saha, Sumit. "A Comprehensive Guide to Convolutional Neural Networks The Eli5 Way." *Medium*, Towards Data Science, 17 Dec. 2018, https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53.