Intro to Image Understanding (CSC420)
Assignment 4

Due Date: Mar 28^{th}, 2025, 11:59 pm
Total: 160 marks

General Instructions:

- You are allowed to work directly with one other person to discuss the questions. However, the implementation and the report should be your own original work; i.e. you should not submit identical documents or codes. If you choose to work with someone else, write your teammate's name on top of the first page of the report.
- Your submission should be in the form of an electronic report (PDF), with the answers to the specific questions (each question separately), and a presentation and discussion of your results. For this, please submit a file called **report.pdf** to MarkUs directly.
- Submit documented codes that you have written to generate your results separately. Please store all of those files in a folder called **assignment4**, zip the folder, and then submit the file **assignment4.zip** to MarkUs. You should include a **README.txt** file (inside the folder) which details how to run the submitted codes.
- Do not worry if you realize you made a mistake after submitting your zip file; you can submit multiple times on MarkUs.
- MarkUs has a file size limit. If your pdf or zip file is larger than the limit, you can try resizing or reducing the resolution of images in your report to reduce the file size. If that does not work, you can split your report into multiple files (e.g. Report_part_1_of_3.pdf, Report_part_2_of_3.pdf, etc.)

Part I: Theoretical Problems (70 marks)

[Question 1] RANSAC (10 marks)

We have two images of a planar object (e.g. a painting) taken from different viewpoints and we want to align them. We have used SIFT to find a large number of point correspondences between the two images and visually estimate that at least 70% of these matches are correct with only small potential inaccuracies. We want to find the true transformation between the two images with a probability greater than 99.5%.

1. **(5 marks)** Calculate the number of iterations needed for fitting a homography.
2. **(5 marks)** Without calculating, briefly explain whether you think fitting an affine transformation would require fewer or more RANSAC iterations and why.

1. Calculate the number of iterations needed for fitting a homography.

Let us begin by defining some variables.

- **p**: desired probability that we get a good sample
- **s**: number of points in a sample
- **N**: number of iterations required
- **e**: probability that a point is an outlier

From this we can make a few observations.

- $1 - e$: probability that a point is an inlier
- $(1 - e)^s$: probability that s points are inliers
- $1 - (1 - e)^s$: probability that at least one of the s points is an outlier.

For N iterations, the probability that all N have s that are outliers is

$$(1 - (1 - e)^s)^N = p$$

, and the probability that at least one is a success is

$$1 - (1 - (1 - e)^s)^N = p.$$

Note that it is equal to p because that is how we defined p earlier.
Now we can solve for N.

$$\begin{aligned} 1 - (1 - (1 - e)^s)^N &= p \\ (1 - (1 - e)^s)^N &= 1 - p \\ N \log(1 - (1 - e)^s) &= \log(1 - p) \\ N &= \frac{\log(1 - p)}{\log(1 - (1 - e)^s)} \end{aligned}$$

As required.

2. Without calculating, briefly explain whether you think fitting an affine transformation would require fewer or more RANSAC iterations and why.

Homography requires 4 points of correspondence, as it has 8 degrees of freedom. Similarly, affine transformations only require 3 points as they have 6 degrees of freedom. The fewer point requirement means s is 3 not 4 in affine transformations. This means, for the same e, the argument of the log in the denominator will be larger for larger s. And thus, the denominator is smaller for larger s. Since p is the same, we conclude that for larger s, N will

be larger too.
Since $s = 3$ in affine transformations, it is clear that N will be smaller than homography, where $s = 4$.

[Question 2] Camera Models (30 marks)

Assume a plane passing through point $\vec{P_0} = [X_0,\ Y_0,\ Z_0]^T$ with normal $\vec{n}$. The corresponding vanishing points for all the lines lying on this plane form a line called the horizon. In this question, you are asked to prove the existence of the horizon line by following the steps below:

1. **(15 marks)** Find the pixel coordinates of the vanishing point corresponding to a line L, passing point $\vec{P_0}$ and going along direction $\vec{d}$.

   Hint: $\vec{P} = \vec{P_0} + t\,\vec{d}$ are the points on line L, and $\vec{p} = \begin{pmatrix} \omega x \\ \omega y \\ \omega \end{pmatrix} = K\,\vec{P} = K \begin{pmatrix} X_0 + t\,d_x \\ Y_0 + t\,d_y \\ Z_0 + t\,d_z \end{pmatrix}$ are pixel coordinates of the same line in the image, and $K = \begin{pmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{pmatrix}$, where f is the camera focal length and (p_x, p_y) is the principal point.

2. **(15 marks)** Prove the vanishing points of all the lines lying on the plane form a line.

   Hint: all the lines on the plane are perpendicular to the plane's normal $\vec{n}$; that is, $\vec{n}\,.\,\vec{d} = 0$, or $n_x\,d_x + n_y\,d_y + n_z\,d_z = 0$

1. Find the pixel coordinates of the vanishing point corresponding to a line L, passing point $\vec{P_0}$ and going along direction $\vec{d}$.

Let's start with the hints. The line L is given by $\vec{P} = \vec{P_0} + t\,\vec{d}$. Its projection on the image plane is given by $\vec{p}$,

$$\begin{aligned} \vec{p} &= \begin{pmatrix} \omega x \\ \omega y \\ \omega \end{pmatrix} = K\,\vec{P} = K \begin{pmatrix} X_0 + t\,d_x \\ Y_0 + t\,d_y \\ Z_0 + t\,d_z \end{pmatrix} \\ &= \begin{pmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X_0 + t\,d_x \\ Y_0 + t\,d_y \\ Z_0 + t\,d_z \end{pmatrix} \\ &= \begin{pmatrix} fX_o + ftd_x + p_xZ_0 + p_xtd_z \\ fY_o + ftd_y + p_yZ_0 + p_ytd_z \\ Z_0 + td_z \end{pmatrix} \end{aligned}$$

In the limit as $t \to \infty$, this will be the vanishing point on the image, as the projection of the real line is limited. Note we divide the x and y components by the homogeneous factor

$\omega = Z_0 + t\,d_z$ to get the x and y values.

$$x = \lim_{t \to \infty} \frac{fX_o + ftd_x + p_x Z_0 + p_x t d_z}{Z_0 + t d_z} = \frac{f d_x + p_x d_z}{d_z}$$
$$y = \lim_{t \to \infty} \frac{fY_o + ftd_y + p_y Z_0 + p_y t d_z}{Z_0 + t d_z} = \frac{f d_y + p_y d_z}{d_z}$$

The expression for the vanishing point does not depend on $\vec{P_0}$ which means this is the vanishing point for all lines of direction $\vec{d}$.
Thus we have shown that the vanishing point for a line L passing through $\vec{P_0}$ along direction $\vec{d}$ is given by

$$(x, y) = (\frac{f d_x + p_x d_z}{d_z}, \frac{f d_y + p_y d_z}{d_z}).$$

And that this is the same point for any line of direction $\vec{d}$.

2. Prove the vanishing points of all the lines lying on the plane form a line.

Again starting with the hint, we observe that all lines in the plane have no component in the direction normal to the plane, $\vec{n}\,.\,\vec{d} = 0$, or $n_x\,d_x + n_y\,d_y + n_z\,d_z = 0$.
We said that $\vec{p}$ is the projection on the image plane of a line L as defined previously. We then found that in the limit as t goes to infinity,

$$\vec{v} = \begin{pmatrix} f d_x + p_x d_z \\ f d_y + p_y d_z \\ d_z \end{pmatrix},$$

we get the homogeneous representation of the vanishing point for the line L.

$$\vec{v} = \begin{pmatrix} f d_x + p_x d_z \\ f d_y + p_y d_z \\ d_z \end{pmatrix} = \begin{pmatrix} d_z x \\ d_z y \\ d_z \end{pmatrix} \rightarrow \begin{cases} d_x = \frac{(x - p_x) d_z}{f} \\ d_y = \frac{(y - p_y) d_z}{f} \\ d_z = d_z \end{cases}$$

We can use this formulation of $\vec{d}$ and the plane equation to solve for the line equation.

$$n_x\,d_x + n_y\,d_y + n_z\,d_z = 0$$
$$n_x \left(\frac{(x - p_x) d_z}{f}\right) + n_y \left(\frac{(y - p_y) d_z}{f}\right) + n_z\,d_z = 0$$
$$\left[n_x \left(\frac{x - p_x}{f}\right) + n_y \left(\frac{y - p_y}{f}\right) + n_z\right] d_z = 0$$
$$n_x \left(\frac{x - p_x}{f}\right) + n_y \left(\frac{y - p_y}{f}\right) + n_z = 0$$
$$n_x (x - p_x) + n_y (y - p_y) + f n_z = 0$$

This is a linear equation in x and y and forms the set of vanishing points for all lines lying in the plane. Thus the vanishing points form a line in the image plane, the horizon line.

$$\vec{d} = \begin{pmatrix} x - p_x \\ y - p_y \\ f \end{pmatrix}$$

is the direction of the horizon line which is shared by all lines on the plane. This vector lies in the plane if the vanishing point (x,y) lies on the horizon line.
Thus we have proved the existence of the horizon line.

[Question 3] Homogeneous Coordinates (30 marks)

Using the homogeneous coordinates:

1. **(15 marks)** (a) Show that the intersection of the 2D line l and l' is the 2D point $p = l \times l'$.
   (here $\times$ denotes the cross product)
2. **(15 marks)** (b) Show that the line that goes through the 2D points p and p' is $l = p \times p'$.

(a) Show that the intersection of the 2D line l and l' is the 2D point $p = l \times l'$.

Let us define the two lines.

$$\vec{l} = \begin{pmatrix} a \\ b \\ c \end{pmatrix} \qquad \vec{l'} = \begin{pmatrix} a' \\ b' \\ c' \end{pmatrix}$$

Then by the definition of cross product.

$$\vec{p} = \vec{l} \times \vec{l'} = \begin{pmatrix} bc' - cb' \\ ca' - ac' \\ ab' - a'b \end{pmatrix}$$

Since we are using homogeneous coordinates, if the inner product is 0, then the point lies in each line.

$$\vec{l}^T \vec{p} = \begin{pmatrix} a & b & c \end{pmatrix} \begin{pmatrix} bc' - cb' \\ ca' - ac' \\ ab' - a'b \end{pmatrix} = abc' - acb' + bca' - bac' + cab' - ca'b$$

$$\begin{aligned} &= abc' - ab'c + a'bc - abc' + ab'c - a'bc \\ &= (abc' - abc') + (a'bc - a'bc) + (ab'c - ab'c) \\ &= 0 \end{aligned}$$

And for $\vec{l'}$.

$$\vec{l'}^T \vec{p} = \begin{pmatrix} a' & b' & c' \end{pmatrix} \begin{pmatrix} bc' - cb' \\ ca' - ac' \\ ab' - a'b \end{pmatrix} = a'bc' - a'cb' + b'ca' - b'ac' + c'ab' - c'a'b$$

$$\begin{aligned} &= a'bc' - a'b'c + a'b'c - ab'c' + ab'c' - a'bc' \\ &= (a'bc' - a'bc') + (a'b'c - a'b'c) + (ab'c' - ab'c') \\ &= 0 \end{aligned}$$

Thus the point $\vec{p} = \vec{l} \times \vec{l'}$ is the intersection of two 2D lines $\vec{l}$ and $\vec{l'}$. Note that we did not need to compute the cross product as it is a property of the cross product to be orthogonal (inner product is 0) to the operands.

(b) Show that the line that goes through the 2D points p and p' is $l = p \times p'$.

Let us define the two points.

$$p = \begin{pmatrix} a \\ b \\ c \end{pmatrix} \qquad p' = \begin{pmatrix} a' \\ b' \\ c' \end{pmatrix}$$

Then by the definition of cross product.

$$l = p \times p' = \begin{pmatrix} bc' - cb' \\ ca' - ac' \\ ab' - a'b \end{pmatrix}$$

Now we show that the two points p and p' are both on this line. Once again, due to the homogeneous coordinates, it is sufficient to show that $l^T p = 0$ and $l^T p' = 0$ to show that the line passes these points.
However, by the properties of cross product, these are given to be the case. Thus the line that goes through the 2D points p and p' is given by $l = p \times p'$.

Part II: Implementation Tasks (90 marks)

[Question 4] Homography (60 marks)

You are given three images `hallway1.jpg`, `hallway2.jpg`, `hallway3.jpg` which were shot with the same camera (i.e. same internal camera parameters), but held at slightly different positions/orientations (i.e. with different external parameters).

`hallway1.jpg`

`hallway2.jpg`

`hallway3.jpg`

Consider the homographies **H**,

$$\begin{bmatrix} \widetilde{w}\widetilde{x} \\ \widetilde{w}\widetilde{y} \\ \widetilde{w} \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

that map corresponding points of one image I to a second image $\widetilde{I}$, for three cases:

A. The right wall of I =`hallway1.jpg` to the right wall of $\widetilde{I}$ =`hallway2.jpg`.
B. The right wall of I =`hallway1.jpg` to the right wall of $\widetilde{I}$ =`hallway3.jpg`.
C. The floor of $\widetilde{I}$ =`hallway1.jpg` to the floor of $\widetilde{I}$ =`hallway3.jpg`.

For each of these three cases:

1. **(10 marks)** Use a Data Cursor to select corresponding points by hand. Select more than four pairs of points. (Four pairs will give a good fit *for those points*, but may give a poor fit for other points.) Also, avoid choosing three (or more) collinear points, since these do not provide independent information. This is trickier for case **C**. Make two **figures** showing the gray-level images of I and $\widetilde{I}$ with a colored square marking each of the selected points. You can convert the image I or $\widetilde{I}$ to gray level using an RGB to grayscale function (or the formula $gray = 0.2989 \times R + 0.5870 \times G + 0.1140 \times B$).
2. **(10 marks)** Fit a homography **H** to the selected points. Include the estimated **H** in the report, and describe its effect using words such as *scale, shear, rotate, translate*, if appropriate. You are not allowed to use any homography estimation function in OpenCV or other similar packages.
3. **(10 marks)** Make a **figure** showing the $\widetilde{I}$ image with red squares that mark each of the selected $(\widetilde{x}, \widetilde{y})$, and green squares that mark the locations of the estimated $(\widetilde{x}, \widetilde{y})$, that is, use the homography to map the selected (x, y) to the $(\widetilde{x}, \widetilde{y})$ space.

4. **(25 marks)** Make a figure showing a new image that is larger than the original one(s). The new image should be large enough that it contains the pixels of the I image as a subset, along with *all* the inverse mapped pixels of the $\widetilde{I}$ image. The new image should be constructed as follows:

   - RGB values are initialized to zero,
   - The red channel of the new image must contain the `rgb2gray` values of the I image (for the appropriate pixel subset only);
   - The blue and green channels of the new image must contain the `rgb2gray` values of the corresponding pixels $(\widetilde{x}, \widetilde{y})$ of $\widetilde{I}$. The correspondence is computed as follows: for each pixel (x, y) in the new image, use the homography **H** to map this pixel to the $(\widetilde{x}, \widetilde{y})$ domain (not forgetting to divide by the homogeneous coordinate), and round the value so you get an integer grid location. If this $(\widetilde{x}, \widetilde{y})$ location indeed lies within the domain of the $\widetilde{I}$ image, then copy the `rgb2gray`'ed value from that $\widetilde{I}(\widetilde{x}, \widetilde{y})$ into the blue and green channel of pixel (x, y) in the new image. (This amounts to an inverse mapping.)
     If the homography is correct and *if the surface were Lambertian** then corresponding points in the new image would have the same values of R,G, and B and so the new image would appear to be gray at these pixels.
   - Based on your results, what can you conclude about the relative 3D positions and orientations of the camera? Give only qualitative answers here. Also, What can you conclude about the surface reflectance of the right wall and floor, namely are they more or less Lambertian? Limit your discussion to a few sentences.

(5 marks) Along with your writeup, hand in the program that you used to solve the problem. You should have a switch statement that chooses between cases **A, B, C**.

* *Lambertian reflectance is the property that defines an ideal "matte" or diffusely reflecting surface. The apparent brightness of a Lambertian surface to an observer is the same regardless of the observer's angle of view. Unfinished wood exhibits roughly Lambertian reflectance, but wood finished with a glossy coat of polyurethane does not, since the glossy coating creates specular highlights. Specular reflection, or regular reflection, is the mirror-like reflection of waves, such as light, from a surface. Reflections on still water are an example of specular reflection.*

For part 1, the plots are provided in the submission for all cases. For part 2, the plots are also provided in the zip file and the estimated **H** and discussion is in this report below. For the program submission, it is also in the zip. The switch case involves inputting a value into the terminal when prompted.

Part 3 Analysis:
Let the subscript describe the case each **H** belongs to.

$$H_A = \begin{bmatrix} 9.02 \cdot 10^{-1} & -3.21 \cdot 10^{-2} & -9.49 \cdot 10^{1} \\ -6.84 \cdot 10^{-2} & 8.86 \cdot 10^{-1} & 3.75 \cdot 10^{2} \\ -3.83 \cdot 10^{-5} & -7.77 \cdot 10^{-5} & 1 \end{bmatrix}$$

$$H_B = \begin{bmatrix} 3.29 \cdot 10^{-1} & -8.31 \cdot 10^{-2} & 4.24 \cdot 10^{2} \\ -8.12 \cdot 10^{-2} & 7.51 \cdot 10^{-1} & 2.48 \cdot 10^{2} \\ -1.64 \cdot 10^{-4} & -1.23 \cdot 10^{-4} & 1 \end{bmatrix}$$

$$H_C = \begin{bmatrix} 5.13 \cdot 10^{-1} & -4.01 \cdot 10^{-1} & 4.02 \cdot 10^{2} \\ -3.21 \cdot 10^{-1} & 7.61 \cdot 10^{-1} & 3.57 \cdot 10^{2} \\ -3.52 \cdot 10^{-4} & -2.25 \cdot 10^{-5} & 1 \end{bmatrix}$$

For H_A, we see that the diagonal elements are (0.9, 0.89, 1), indicating a modest contraction along the x and y axes. The bottom row is very close to (0, 0, 1) with values on the order of 10^{-5}. This indicates that the mapping is indeed affine. The translation components are -94.9 in x and 375 in y, indicating that the right wall is translated a lot in the target. The off diagonal entries, -0.03 and -0.068 are really small indicating minor rotation and shearing effects. This the main effect of H_A is translation and scaling, with minor rotation.
For H_B, we see that the diagonal elements are (0.329, 0.75, 1), indicating that the x scaling is much smaller than the y scaling. This suggests that features in the x direction are compressed more than in the y. Once again, the bottom row is very close to (0, 0, 1) with values on the order of 10^{-4}. This indicates that the mapping is indeed affine. The translation components are 424 in x and 248 in y. Once again, this shows that the mapped region is shifted to a new part in the target. The off diagonal entries, -0.08 and -0.08 are really small indicating minor rotation and shearing effects. These are higher than H_A but still quite small. This the main effect of H_b is translation and scaling, with minor rotation.
For H_c, we see that the diagonal elements are (0.51, 0.76, 1), indicating that the x scaling is somewhat smaller than the y scaling. This suggests that features in the x direction are compressed a bit more than in the y. Once again, the bottom row is very close to (0, 0, 1) with values on the order of 10^{-4}. This indicates that the mapping is indeed affine. The translation components are 402 in x and 357 in y. Once again, this shows that the mapped region is shifted to a new part in the target. The off diagonal entries, -0.40 and -0.32 are way larger than the other cases. This indicates a more pronounced rotation effect is occurring. This makes sense as the floor is oriented differently in both images. These are higher than H_A but still quite small. This the main effect of H_b is translation and scaling, with minor rotation.

Part 4 Analysis:
The images are available in the zip file.
Case A:
The overlap of the red and cyan channels appears rather consistent on the right wall, indicating that the camera moved only slightly between hallway1.jpg and hallway2.jpg. There is also some horizontal shift and a small tilt, but overall the perspective change is mild.
The overlapping region on the wall looks pretty gray, implying that the wall behaves Lambertian. The slight fringing in the image can be attributed to small viewpoint differences or lighting changes.
Case B:
Compared to Case A, the transformation is more pronounced. The composite shows a bigger tilt, so the camera was positioned differently or angled more when hallway3.jpg was taken. The right wall still largely overlaps, but the distortion is more obvious than in Case A.
The main portion of the wall remains mostly gray where the planes align well, again indi-

cating a mostly Lambertian surface. However, at edges or near strong reflections, there are some color mismatches.
Case C:
Here, the camera is viewing the floor plane from different angles, causing a stronger perspective effect. The camera probably changed both its horizontal position and its tilt/pitch relative to the floor.
The floor has more noticeable reflections and varied lighting, so parts of the composite do not blend to perfect gray. This suggests the floor is less purely Lambertian which is totally expected. The overlap is still reasonably gray in some regions, but overall shows more color fringing than the right wall cases.

[Question 5] Mean Shift Tracking (30 marks)

In tutorial 10, we learned about the mean shift and cam shift tracking. In this question, we first attempt to evaluate the performance of mean shift tracking in a single case and will then implement a small variation of the standard mean shift tracking. For both parts you can use the attached short video `KylianMbappe.mp4` or, alternatively, you can record and use a short (2-3 second) video of yourself. You can use any OpenCV (or other) functions you want in this question.

1. **(20 marks)** Performance Evaluation

    - Use the Viola-Jones face detector to detect the face on the first frame of the video. The default detector can detect the face in the first frame of the attached video. If you record a video of yourself, make sure your face is visible and facing the camera in the first frame (and throughout the video) so the detector can detect your face in the first frame.
    - Construct the **hue** histogram of the detected face on the first frame using appropriate **saturation** and **value** thresholds for masking. Use the constructed **hue** histogram and mean shift tracking to track the bounding box of the face over the length of the video (from frame #2 until the last frame). So far, this is similar to what we did in the tutorial.
    - Also, use the Viola-Jones face detector to detect the bounding box of the face in each video frame (from frame #2 until the last frame).
    - Calculate the intersection over union (IoU) between the tracked bounding box and the Viola-Jones detected box in each frame. Plot the IoU over time. The x axis of the plot should be the frame number (from 2 until the last frame) and the y axis should be the IoU on that frame.
    - In your report, include a sample frame in which the IoU is large (e.g. over 50%) and another sample frame in which the IoU is low (e.g. below 10%). Draw the tracked and detected bounding boxes in each frame using different colors (and indicate which is which).

- Report the percentage of frames in which the IoU is larger than 50%.
- Look at the detected and tracked boxes at frames in which the IoU is small ($< 10\%$) and report which (Viola-Jones detection or tracked bounding box) is correct more often (we don't need a number, just eyeball it). Very briefly (1-2 sentences) explain why that might be.

2. **(10 marks)** Implement a Simple Variation

   - In the examples in Tutorial 10 (and the previous part of this question) we used a **hue** histogram for mean shift tracking. Here, we implement an alternative in which a histogram of gradient direction values is used instead.
   - After converting to grayscale, use blurring and the Sobel operator to first generate image gradients in the x and y directions (I_x and I_y). You can then use `cartToPolar` (with `angleInDegrees=True`) to get the gradient magnitude and angle at each frame. You can use 24 histogram bins and [0,360] (i.e. not [0,180]) directions.
   - When constructing **hue** histograms, we thresholded **saturation** and **value** channels to create a mask. Here, you can threshold the gradient magnitude to create a mask. For example, you can mask out pixels in the region of interest in which the gradient magnitude is less than 10% of the maximum gradient magnitude in the RoI.
   - Calculate the intersection over union (IoU) between the tracked bounding box and the Viola-Jones detected box in each frame. Plot the IoU over time. The x axis of the plot should be the frame number (from 2 until the last frame) and the y axis should be the IoU on that frame.
   - In your report, include a sample frame in which the IoU is large (e.g. over 80%) and another sample frame in which the IoU is low (e.g. below 50%). Draw the tracked and detected bounding boxes in each frame using different colors (and indicate which is which).
   - Report the percentage of frames in which the IoU is larger than 50%.

The plots of the IoU over time for both hue and gradient are in the zip file.

The percentage of frames with IoU $> 50\%$: 59.46% for hue-based tracking. Figure 1 below includes the sample frames for low and large IoU, with the green being the tracked, and the red being the detected.
In most low-IoU frames, the tracked box (green) is more accurate than the Viola–Jones detection (red). This is probably because the tracker maintains consistency in time, while Viola–Jones fails to detect the face accurately due to changes in lighting, pose, or small blockages.

(a) Sample hue image with low IoU.

(b) Sample hue image with high IoU.

Figure 1: Hue-based IoU plots..

The percentage of frames with IoU $> 50\%$ for the gradient-based tracking is 21.62%. Figure 2 below includes the sample frames for low and large IoU, with the green being the tracked and the red being the detected.

(a) Sample gradient image with low IoU.

(b) Sample gradient image with high IoU.

Figure 2: Gradient-based IoU plots..