Homework 5: Computer Vision

EECS/ME/BioE C106A/206A Introduction to Robotics

Fall 2023

Note: This problem set includes programming components, but you will be submitting screenshots of your results, so there is no autograder for this assignment. Submit a pdf of your work for all problems to Gradescope assignment.

Problem 1: Two-View Triangulation

Consider two cameras with reference frames $\{1\}$ and $\{2\}$ respectively. As always, the reference frame of each camera is such that the $X - Y$ plane is parallel to the image plane and the Z-axis points in the direction of viewing.

Assume we know the relative transform $g_{21} = (R, T) \in SE(3)$. Additionally, assume the cameras are calibrated and normalized, so that the camera matrix K is the identity.

Both cameras are looking at the same point p in 3D space, which has unknown coordinates $X_1 \in \mathbb{R}^3$ in frame $\{1\}$ and $X_2 \in \mathbb{R}^3$ in frame $\{2\}$. We observe their image coordinates x_1 and x_2, written in 2D homogeneous coordinates.

(a) Write down an expression relating x_1 to X_1 in terms of an unknown depth λ_1. Do the same for camera 2.

(b) Write down an expression for X_2 in terms of X_1.

(c) Find a method for solving for X_1 in terms of the known quantities $R \in SO(3), T \in \mathbb{R}^3, x_1, x_2$. Can you deal with the case when the image measurements x_1, x_2 are corrupted by some small (white, zero mean, Gaussian) noise?

Hint: Eliminate X_1 and X_2 from your expressions, and try to find only the unknown depths λ_1 and λ_2. Then, use these depths to recover X_1.

Problem 2: Epipolar Ambiguities and Structure from Motion

Consider a similar set up as in the previous problem, with two calibrated, normalized cameras, where the transform $g_{21} = (R, T)$ between them is *not* known. Recall that for such a system, we define $E = \hat{T}R \in \mathbb{R}^{3\times 3}$ to be the *essential matrix*. The essential matrix imposes the *epipolar constraint*, which is that whenever x_1 and x_2 are the (homogeneous) image coordinates of the *same* point, then they must satisfy

$$x_2^T E x_1 = 0$$

Such a pair of image points x_1, x_2 that correspond to the same point in 3D space viewed from two different cameras are called *corresponding points*. In this problem, we consider the problem of recovering the relative poses between cameras in a multi-camera setup when we are given a number of corresponding-point pairs.

It turns out that 8 pairs of corresponding points $(x_1^{(1)}, x_2^{(1)}), \cdots (x_1^{(8)}, x_2^{(8)})$ in general position are enough to compute a candidate essential matrix $\tilde{E}$. Each such pair gives us an equation of the form

$$x_2^{(i)T} E x_1^{(i)} = 0 \tag{1}$$

where the x's are all known. We additionally have the constraint that E should be of the correct form to be written as $\hat{T}R$. i.e. we should be able to write it as the product of a cross product matrix $\in \mathfrak{so}(3)$ and a rotation matrix $\in SO(3)$. We can then solve this system of equations for a nonzero 3×3 matrix E that satisfies this set of constraints. See chapter 5 from *An Invitation to 3D Vision (Ma, Soatto, Kosecka, Sastry)* for the full details.

(a) Show that we can only recover E up to a scale factor. In particular, show that if $\tilde{E}$ is a matrix that satisfies all the required constraints, then so is $c\tilde{E}$ for any real number c.

Remark: We can in fact conclude that this ambiguity can be attributed to an unknown scale factor on the translation vector T between the two frames. This means that although we can decompose a computed essential matrix E into rotational and translational components $(R, \tilde{T})$, we can only recover the original translation T up to a scale factor. Typically then, we restrict ourselves to finding a $\tilde{T}$ such that $\|\tilde{T}\| = 1$.

(b) Say we have a system of 3 cameras with reference frames $\{1\}, \{2\}$ and $\{3\}$ respectively, and we are able to recover the transforms $(R_{12}, \tilde{T}_{12}), (R_{23}, \tilde{T}_{23})$ and $(R_{13}, \tilde{T}_{13})$ using point correspondances, where each $\tilde{T}_{ij}$ has norm 1. So there are unknown, nonzero scale factors λ_{ij} such that the true $T_{ij} = \lambda_{ij}\tilde{T}_{ij}$. If we could find the three scalars $\lambda_{12}, \lambda_{23}, \lambda_{13}$ then we would have fully recovered the relative poses between the various cameras. Show that in this setting, we can only recover the λ_{ij}'s up to a single scaling factor.

(c) Consider the same setup as part (c), but now the translation T_{12} is known exactly (i.e. λ_{12} is known). Show that now, all λ_{ij}'s can be recovered and the relative poses between the cameras can be found.

Problem 3: Planar Motion Models

For feature tracking algorithms we often assume that the motion of points in the image when restricted to a small window can be approximated through different *transformations* of varying levels of complexity. These assumptions may only hold for a small window, but for an appropriate object, there exist motions in 3D space such that these transformations are accurate over the entire image. In this question, we will determine what those motions are.

Assume we are only concerned with the motion of image points corresponding to an object where all points in the object have the same $z-$coordinate z_o relative to the camera frame (ie. all world points of interest lie in some plane parallel to the $x-y$ plane which passes through the point $\begin{bmatrix}0 & 0 & z_o\end{bmatrix}^T$)

(a) Define $h(x)$ as a function which maps an image point to its new location after the corresponding world point undergoes a rigid motion. Let's consider a scenario where we measure image motion $h(x)$ and we notice that each point on the image corresponding to our object translates by the same Δx. More concretely, $h(x) = x + \Delta x$. Prove that a rigid body motion $R = I$ and $T = \begin{bmatrix}a & b & 0\end{bmatrix}^T$ applied to our object corresponds to this $h(x)$.

(b) Now let's consider a scenario where $h(x) = Ax + d$ for image points corresponding to our object. Prove that a rigid body motion $R = R_Z(\theta)$ with arbitrary T applied to our object corresponds to this motion.

Problem 4: Geometry in the time of (deep) learning

Some autonomous car companies such as Tesla and Wayve are experimenting with *end-to-end driving*, where a deep neural network takes in all sensor inputs and directly predicts the future trajectory of the car. Mathematically, this can be written as a map from the sensor data o_t to a trajectory $\tilde{X}$:

$$f(o_t) = [s_{t+1}, ..., s_{t+H}] := \tilde{X}_t \in \mathbb{R}^{2\times H} \tag{2}$$

where H is the control time horizon and $s_t = [x_t, y_t]^T$ is the desired location of the car at time t on the road plane.
You might be tempted to think that in this scenario we can throw out all our fancy geometry, but not so fast! It still plays a critical role in *analyzing* the behavior of our neural network.

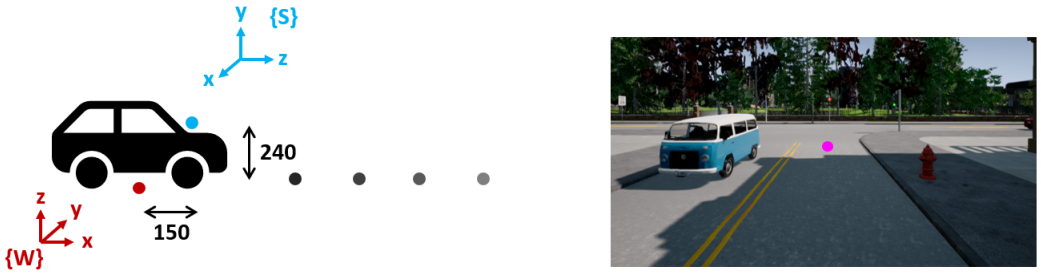

Figure 1: Left: The world frame is located on the road beneath the center of the car at the initial time. A front-facing camera is mounted in on the front of the car, and the relative position and orientation of the camera and world frames are shown. Right: For part (b), the location where the car will come to a stop is shown with a pink dot.

a) Let's say your car has a forward-facing RGB camera and you want to visualize the car's intended trajectory by plotting the predicted future points on the image. You are given:

   (a) The future trajectory in the world coordinate frame {W}

   (b) The camera calibration matrix K_f

   See the figure for an illustration of the camera frame and world frames. Fill out the `world_pt_to_pixel` function in the provided notebook to return the correct pixel location for each trajectory point. Submit a screenshot of the trajectory plot. *Hint: see section 3.3 of An Invitation to 3D Vision*

b) You got a nice new high-res camera for your car, but your old camera parameters aren't working anymore! You know the new camera has square pixels and no skew distortion, but you're not sure about the focal length f. You look through some data you've collected and notice that at time $t = 5$ the car will stop right behind the end of the lane marking. You decide to plot the future trajectory at $t = 0$ and use the known future position to find f and (roughly) calibrate your camera.

   Fill out the `world_pt_f_to_pixel` function in the provided notebook to return the pixel locations of the trajectory for a particular value of f. Use the visualization to tune f. Submit a screenshot of the final trajectory plot.

c) You're now out on the road using your calibrated camera to check the safety of your car's trajectory. Use the f from part (b) to visualize the predicted trajectory in the notebook. Is this trajectory safe?