Camera Sensor

A. Giavaras

Contents

1 Camera Sensor

The camera sensor is one of the primary sensors in a vehicle's sensor suite. This is because the camera is a rich sensor that captures incredible detail about the environment around the vehicle. However, it requires extensive processing to make use of the information that is available in that image.

In this section, we will highlight why the camera is a critical sensor for autonomous driving. We will then briefly introduce the concept of image formation and present the pinhole camera model which captures the essential elements of how a camera works in a simple and elegant manner. We'll then show you an example of a historic camera design which used the pinhole principle to create some of the earliest images ever recorded.

Of all the common self-driving car sensors, the camera is the sensor that provides the most detailed appearance information from objects in the environment. Appearance information is particularly useful for scene understanding tasks such as object detection, segmentation and identification. Appearance information is what allows us to distinguish between road signs or traffic lights states, to track turn signals and resolve overlapping vehicles into separate instances. Because of its high resolution output, the camera is able to collect and provide orders of magnitude, more information than other sensors used in self-driving while still being relatively inexpensive. The combination of high valued appearance information and low cost make the camera an essential component of our sensor suite.

1.1 The Pinhole Camera Model

Let us see how the camera manages to collect this huge amount of information. A camera is a passive external receptive sensor. It uses an imaging sensor to capture information conveyed by light rays emitted from objects in the world. This was originally done with film but nowadays we use rather sophisticated silicon chips to gather this information. Light is reflected from every point on an object in all directions, and a portion of these rays travel towards the camera sensor.

Look at the car's reflected rays collected by our imaging surface. Do you think we will get a good representation of the car on the image sensor from this ray-pattern? Unfortunately, no. Using this basic open sensor camera design, we will end up with blurry images because our imaging sensor is collecting light rays from multiple points on the object at the same location on the sensor. The solution to our problem is to put a barrier in front of the imaging sensor with a tiny hole or aperture in its center. The barrier allows only a small number of light rays to pass through the aperture, reducing the blurriness of the image. This model is called the **pinhole camera model** and describes the relationship between a point in the world and it's corresponding projection on the image plane.

The two most important parameters in a pinhole camera model are the distance between the pinhole and the image plane which we call the focal length and is typically denoted with f. The focal length defines the size of the object projected onto the image and plays an important role in the camera focus when using lenses to improve camera performance.

Remark 1.1. **Focal Lenght** f

Specifically, we define the focal length f as the distance between the camera and the image coordinate frames along the z-axis of the camera coordinate frame.

The coordinates of the center of the pinhole, (c_u, c_v), which we call the camera center, these coordinates to find the location on the imaging sensor that the object projection will inhabit.

Although the pinhole camera model is very simple, it works surprisingly well for representing the image creation process. By identifying the focal length and the camera's center for a specific camera configuration, we can mathematically describe the location that a ray of light emanating from an object in the world will strike the image plane. This allows us to form a measurement model of image formation for use in state estimation and object detection.

Remark 1.2. **Some History**

A historical example of the pinhole camera model is the camera obscura, which translates to dark room camera in English. Historical evidence shows that this form of imaging was discovered as early as 470 BC in ancient China and Greece. It's simple construction with a pinhole aperture in front of an imaging surface makes it easy to recreate on your own, and is in fact a safe way to watch solar eclipse if you're so inclined.

Nowadays cameras allow us to collect extremely high resolution data. They can operate in low-light conditions or at a long range due to the advanced lens optics that gather a large amount of light and focus it accurately on the image plane. The resolution and sensitivity of camera sensors continues to improve, making cameras one of the most ubiquitous sensors on the planet. How many cameras do you think you own? You'd be surprised if you try to count them all. You'll have cameras in your phones, in your car, on your laptop, they are literally everywhere and in every device we own today.

These advances are also extremely beneficial for understanding the environment around a self-driving car. Cameras specifically designed for autonomous vehicles need to work well in a wide range of lighting conditions and in distances to objects. These properties are essential to driving safely in all operating conditions.

1.2 Summary

This was an introductory section. We discussed the usefulness of the camera as a sensor for autonomous driving. We also saw the pinhole camera model in its most basic form, which we'll use to construct algorithms for visual perception. In the next section, we will describe how an image is formed, a process referred to as projective geometry, which relates objects in the world to their projections on the imaging sensor.

1.3 Questions

1. Describe the pinhole camera model. Why is it useful?

1.4 Assignements

1. Using OpenCV read and display an image.

2 Camera Projective Geometry

In this section, you will learn how to model the cameras projective geometry through the coordinate system transformation. These transformations can be used to project points from the world frame to the image frame, building on the pinhole camera model from section 1.1.

You will then model these transformations using matrix algebra and apply them to a 3D point to get it's 2D projection onto the image plane. Finally, you will learn how camera 2D images are represented in software. Equipped with the projection equations in image definitions, you will then be able to create algorithms for detecting objects in 3D and localizing the self-driving car later on in the course.

3 Problem Definition

First, let's define the problem we need to solve. Let's start with a point $\mathbf{O}_{world}$ defined at a particular location in the world coordinate frame. We want to project this point from the world frame to the camera image plane. Light travels from the $\mathbf{O}_{world}$ on the object through the camera aperture to the sensor surface. You can see that our projection onto the sensor surface through the aperture results in flipped images of the objects in the world. To avoid this confusion, we usually define a virtual image plane in front of the camera center. Let's redraw our camera model with this sensor plane instead of the real image plane behind the camera lens. We will call this model the simplified camera model, and need to develop a model for how to project a point from the world frame coordinates x, y, z to, image coordinates u, v.

We begin by defining the following characteristics of the cameras that are relevant to our problem. First, we select a world frame in which to define the coordinates of all objects and the camera. We also define the camera coordinate frame as the coordinate frame attached to the center of our lens aperture known as the optical sensor. We can define a translation vector and a rotation matrix to model any transformation between a world coordinate frame and another, and in this case, we'll use the world coordinate frame and the camera coordinate frame. We refer to the parameters of the camera pose as the extrinsic parameters, as they are external to the camera and specific to the location of the camera in the world coordinate frame. We define our image coordinate frame as the coordinate frame attached to our virtual image plane emanating from the optical center. The image pixel coordinate system however, is attached to the top left corner of the virtual image plane. So we'll need to adjust the pixel locations to the image coordinate frame. Next, we define the focal length f as the distance between the camera and the image coordinate frames along the z-axis of the camera coordinate frame. Finally, our projection problem reduces to two steps.

1. Project from the world to the camera coordinates
2. Project from the camera coordinates to the image coordinates

We can then transform image coordinates to pixel coordinates through scaling and offset. We now have the geometric model to allow us to project a point from that world frame to the image coordinate frame, whenever we want.

3.1 Mathematical Formulation

Let us formulate the mathematical tools needed to perform this projection using linear algebra. First, we begin with the transformation from the world to the camera coordinate frame. This is performed using the rigid body transformation matrix $\mathbf{T}$, which has $\mathbf{R}$ and t in it. The next step is to transform camera coordinates to image coordinates. To perform this transformation, we define the matrix $\mathbf{K}$ as a three-by-three matrix.

$$\mathbf{K} = \begin{bmatrix} f & 0 & u_0 \\ 0 & f & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1}$$

This matrix depends on camera intrinsic parameters, which means it depends on components internal to the camera such as the camera geometry and the camera lens characteristics. Since both transformations are just matrix multiplications, we can define a matrix **P**

$$\mathbf{P} = \mathbf{K}[\mathbf{R}|t] \tag{2}$$

This matrix transforms from the world coordinate frame all the way to the image coordinate frame. The coordinates of point $\mathbf{O}_{world}$ can now be projected to the image plane via the equation

$$\mathbf{O}_{image} = \mathbf{P}\mathbf{O}_{world} \tag{3}$$

So, let's see what we're still missing to compute this equation. When we expect the matrix dimensions, we noticed that the matrix multiplication cannot be performed. To remedy this problem, we transform the coordinates of the point **O** into homogeneous coordinates, and this is done by adding a one at the end of the 3D coordinates.

Remark 3.1. **Homogeneous Coordinates**

The point geometric primitive can be represented using homogeneous coordinates $\tilde{\mathbf{x}}$. Consider for example a 2D point $\mathbf{x} = (x_1, x_2)$ this can be written as $\tilde{\mathbf{x}} = (\tilde{x}_1, \tilde{x}_2, \tilde{w}) \in P^2$, where vectors that differ only by scale are considered to be equivalent. The $P^2 = R^3 - (0, 0, 0)$ is called the 2D projective space. A homogeneous vector can be converted back into an inhomogeneous vector by dividing through the last element $\tilde{w}$:

$$\tilde{\mathbf{x}} = (\tilde{x}_1, \tilde{x}_2, \tilde{w}) = \tilde{w}(x_1, x_2, 1) = \tilde{w}\mathbf{x} \tag{4}$$

Homogeneous [points whose last elelemt is $\tilde{w} = 0$ are called ideal points or points at infinity and do not have an equivalent inhomogeneous representation.

So, now the dimensions work and we're all ready to start computing our projections. Now, we need to perform the final step, transforming the image coordinates to pixel coordinates. We do so by dividing x and y by z to get homogeneous coordinates in the image plane.

This is the basic camera projection model. In practice, we usually model more complex phenomena such as non-square pixels, camera access skew, distortion and non unit aspect ratio. Luckily, this only changes the camera **K** matrix, and the equations above can be used as is with a few additional parameters.

Now that we have formulated the coordinates of projection of a 3D point onto the 2D image plane, we want to define what values go into the coordinates in a

2D color image. We will start with a grayscale image. We first define a width N and a height M of an image, as the number of rows and columns the image has. Each point in 3D projects to a pixel on the image defined by the u, v coordinates we derived earlier. Zooming in, we can see these pixels is a grid. In grayscale, brightness information is written in each pixel as an unsigned eight bit integer. Some cameras can produce unsigned 16-bit integers for better quality images. For color images, we have a third dimension of value three we call depth. Each channel of this depth represents how much of a certain color exists in the image.

Many other color representations are available, but we will be using the RGB representation, so red green and blue.

In conclusion, an image is represented digitally as an $M \times N \times 3$ array of pixels, with each pixel representing the projection of a 3D point onto the 2D image plane.

4 Summary

So, in this section, we discussed how to project 3D points in the world coordinate frame to 2D points in the image coordinate frame. You saw that the equations that perform this projection rely on camera intrinsic parameters as well as on the location of the camera in the world coordinate frame.

This projection model is used in every visual perception algorithm we develop, from object detection to derivable space estimation. Finally, we saw that images are represented in software as an array representing pixel locations. In the next section, we will discuss how to tailor the camera model to a specific camera by computing its intrinsic and extrinsic camera parameters through a process known as camera calibration.

5 Camera Calibration

References

[1] SAE *Taxonomy of Driving* $https : //www.sae.org/standards/content/j3016\_201806/?PC = DL2BUY$

[2] SAE *SAE J3016 Taxonomy and Definitions Document* $https : //drive.google.com/open?id = 1xtOqFVJvOElXjXqf4RAwXZkI\_EwbxFMg$

[3] Åström K. J., Murray R. M. *Feedback Systems. An Introduction for Scientists and Engineers*

[4] Philip , Florent Altche1, Brigitte dAndrea-Novel, and Arnaud de La Fortelle *The Kinematic Bicycle Model: a Consistent Model for Planning Feasible Trajectories for Autonomous Vehicles?* HAL Id: hal-01520869, https://hal-polytechnique.archives-ouvertes.fr/hal-01520869

[5] Marcos R. O., A. Maximo *Model Predictive Controller for Trajectory Tracking by Differential Drive Robot with Actuation constraints*