Image Processing & Projective geometry

Arunkumar Byravan

Partial slides borrowed from Jianbo Shi & Steve Seitz
Color spaces

RGB – Red, Green, Blue

HSV – Hue, Saturation, Value
Why HSV?

• HSV separates *luma*, or the image intensity, from *chroma* or the color information
  • Different shades of a color have same hue, different RGBs

• **Advantage:** Robustness to lighting conditions, shadows etc
  • Easy to use for color thresholding!
  • Fast conversion from RGB to HSV (*Python: colorsys*)

• Other relevant color spaces: YCbCr, Lab
HSV example

- GIMP
MPC – Last lecture

• Repeat:
  • Pick a set of controls (Ex: linear velocity, steering angle)
  • Simulate/Rollout using internal model (for time T)
  • Compute error (Ex: distance from desired path, desired fixed point etc.)
  • Choose controls that minimize error
  • Execute control for T’ << T

• Key questions:
  • How to generate rollouts?
  • How to measure error?
MPC - Racecar

• Rollouts – Image templates
  • Pixel tracks of potential paths taken by the car
  • How to generate them?

• Errors – Distance from track in image
  • How to measure error?
  • Other error metrics:
    • Distance to target point
    • Parameterized error (line, spline etc)
How to generate templates?

• Heuristic approach:
  • Generate arcs of varying curvature
  • Associate with a control (linearly interpolate controls & ”invent” a mapping)
How to generate samples?

• Geometric approach – use motion model

• Generate rollouts from kinematic model based on controls
  • Linearly interpolate controls, rollout a trajectory for fixed horizon “T”

• “Project” rollouts onto image to generate templates
  • Imagine how each rollout would look like when seen from the camera
How to generate templates?

Rollouts from the kinematic motion model, points in frame of car, $z = 0$

How do we generate image templates from these rollouts?

• Projective geometry!
Camera Extrinsicss

Origin frame for rollouts

Red is camera frame, white is rollout origin frame
Camera Extrinsics

- We have 3D points in robot frame (white)
- Transform points to camera frame through extrinsics:

\[
\begin{pmatrix}
  x' \\
  y' \\
  z'
\end{pmatrix} = \begin{bmatrix}
  R & t \\
  0 & 1
\end{bmatrix}
\begin{pmatrix}
  x \\
  y \\
  z
\end{pmatrix}
\]

- \( R \) = 3x3 rotation matrix
- \( t \) = 3x1 translation vector
- For racecar, extrinsics can be measured (and is constant)
Camera Extrinsics

• Extrinsics allow us to transform 3D points to camera frame of reference

• We need to figure out how to get the image of these points as seen by the camera
Image formation process
Pinhole camera model

- Add a barrier to block off most of the rays
  - This reduces blurring
  - The opening known as the **aperture**
  - How does this transform the image?
Pinhole camera model

- Pinhole model:
  - Captures **pencil of rays** – all rays through a single point
  - The point is called **Center of Projection (COP)**
  - The image is formed on the **Image Plane**
  - **Effective focal length** \( f \) is distance from COP to Image Plane
Homemade pinhole camera

Why so blurry?

http://www.debevec.org/Pinhole/
Camera with Lens

Slide from Jianbo Shi
Digital camera

FIGURE 2.17  (a) Continuous image projected onto a sensor array. (b) Result of image sampling and quantization.

Slide from Jianbo Shi
Bayer grid

Estimate RGB at ‘G’ cells from neighboring values

http://www.cooldictionary.com/words/Bayer-filter.wiki
Projection from 3D to 2D
3D point projection (Metric space)

\((X, Y, Z) \rightarrow (u_{\text{ccd}}, v_{\text{ccd}}) = (f_m \frac{X}{Z}, f_m \frac{Y}{Z})\)

2D projection onto CCD plane

\(f_m: \) Focal length in meter
3D point projection (Metric space)

3D point $\mathbf{X} = (X, Y, Z)$ is projected onto the 2D CCD plane as

$$(u_{ccd}, v_{ccd}) = (f_m \frac{X}{Z}, f_m \frac{Y}{Z})$$

2D projection onto CCD plane

Projection plane and focal length in meter.
3D point projection (Metric space)

Focal length in meter

\[(X, Y, Z) \rightarrow (u_{ccd}, v_{ccd}) = (f_m \frac{X}{Z}, f_m \frac{Y}{Z})\]

2D projection onto CCD plane
3D point projection (Pixel space)

\[ \frac{u_{ccd}}{w_{ccd}} = \frac{u_{\text{img}} - p_x}{w_{\text{img}}} \]
\[ \frac{v_{ccd}}{w_{ccd}} = \frac{v_{\text{img}} - p_y}{h_{\text{img}}} \]

\[ u_{\text{img}} = u_{ccd} \frac{w_{\text{img}}}{w_{ccd}} + p_x \]
\[ v_{\text{img}} = v_{ccd} \frac{h_{\text{img}}}{h_{ccd}} + p_y \]

Image principal point \((p_x, p_y)\)
3D point projection (Pixel space)

Projection plane

\[(X, Y, Z) \rightarrow (u_{ccd}, v_{ccd}) = \left(\frac{f_m X}{Z}, \frac{f_m Y}{Z}\right)\]

\[u_{img} = u_{ccd} \frac{w_{img}}{w_{ccd}} + p_x = f_m \frac{w_{img} X}{w_{ccd} Z} + p_x\]

Focal length in pixel

\[v_{img} = v_{ccd} \frac{h_{img}}{h_{ccd}} + p_y = f_m \frac{h_{img} Y}{h_{ccd} Z} + p_y\]

Focal length in pixel

Slide from Jianbo Shi
3D point projection (Pixel space)

\[(X, Y, Z) \rightarrow (u_{ccd}, v_{ccd}) = \left( f_m \frac{X}{Z}, f_m \frac{Y}{Z} \right)\]

\[u_{img} = u_{ccd} \frac{w_{img}}{w_{ccd}} + p_x = f_m \frac{f_x}{w_{ccd}} \frac{X}{Z} + p_x\]

Focal length in pixel

\[v_{img} = v_{ccd} \frac{h_{img}}{h_{ccd}} + p_y = f_m \frac{f_y}{h_{ccd}} \frac{Y}{Z} + p_y\]

Focal length in pixel

Slide from Jianbo Shi
3D point projection (Pixel space)

\[(X,Y,Z) \rightarrow (u_{\text{img}}, v_{\text{img}}) = \left( \frac{W_{\text{img}}}{w_{\text{ccd}}} \frac{X}{Z}, \frac{W_{\text{img}}}{h_{\text{ccd}}} \frac{Y}{Z}, f_{m} \right)\]
Homogeneous coordinates

A point in Euclidean space \((\mathbb{R}^2)\) can be represented by a homogeneous representation in Projective space \((\mathbb{P}^2)\) (3 numbers).

\[
(x, y) \rightarrow (x, y, 1) = f(x, y, 1) = \lambda(x, y, 1)
\]

2D point \(=\) 3D ray \(\lambda(x, y, 1)\)
Homogeneous coordinates

2D point $= 3D$ ray

$\lambda(x, y, 1) = (X, Y, Z)$ : 3D point lies in the 3D ray passing 2D image point.

Homogeneous coordinate
3D point projection (Pixel space)

\[ (X, Y, Z) \rightarrow (u_{\text{ccd}}, v_{\text{ccd}}) = \left( f_m \frac{X}{Z}, f_m \frac{Y}{Z} \right) \]

\[ u_{\text{img}} = f_x \frac{X}{Z} + p_x \quad v_{\text{img}} = f_y \frac{Y}{Z} + p_y \]

\[
\lambda \begin{bmatrix} u_{\text{img}} \\ v_{\text{img}} \end{bmatrix} = \begin{bmatrix} f_x & p_x \\ f_y & p_y \\ 1 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}
\]

Homogeneous representation
Camera intrinsic parameter

\[
\begin{bmatrix}
X \\
Y \\
Z
\end{bmatrix} = \begin{bmatrix}
f_x & p_x \\
0 & 1 \\
0 & 0
\end{bmatrix}
\begin{bmatrix}
\lambda u_{\text{img}} \\
\lambda v_{\text{img}} \\
1
\end{bmatrix}
\]

: metric space to pixel space

Camera intrinsic parameter

Camera intrinsic parameter

Slide from Jianbo Shi
Putting it all together – Generating templates

- Procedure:
  - Generate rollouts based on kinematic car model (robot frame)
  - Transform points to camera frame based on **camera extrinsics**
  - Project points to pixel space using **camera intrinsics**

\[
\begin{align*}
\begin{pmatrix}
\hat{x}' \\
\hat{y}' \\
\hat{z}'
\end{pmatrix}
&= 
\begin{bmatrix}
R & t \\
0 & 1
\end{bmatrix}
\begin{pmatrix}
x \\
y \\
z
\end{pmatrix} \\
\begin{pmatrix}
\hat{u} \\
\hat{v}
\end{pmatrix}
&= K
\begin{pmatrix}
\hat{x}' \\
\hat{y}' \\
\hat{z}'
\end{pmatrix} \\
\end{align*}
\]

**Extrinsics**
- Need to be measured for racecar (Approx values in /tf)

**Intrinsics**
- Intrinsics fixed for a camera (for racecar: /camera/color/camera_info)
Rollout templates

Rollouts from kinematic car model

Projected rollouts
Measuring error (for MPC)

• Template matching using convolution
  • Find template that best matches masked track, choose it

• Issues?
Measuring error (Set-point error)

- Choose set point in image (similar to PID)
  - Find template that gets you closest to set point, choose it
Measuring error (Set-point + Direction error)

• Choose set point in image along with heading (based on track)
  • Find template that gets you closest to set point while oriented correctly
  • Keep track of heading in templates
Measuring error (3D error)

- Instead of generating pixelized templates, project masked track (or set point) back to 3D

- How?
  - Each pixel corresponds to ray in 3D
  - We know that all pixels on track lie on ground plane (known)
  - Solve for ray-plane intersection

- Advantage: Reason in 3D!
Measuring error (fancy error metric)

• Fit a line or curve to pixel/3D track points & your rollouts
  • Compare the errors in parametric space (line / curve co-efficients)
Focal length

Slide from Jianbo Shi
Focal length
Focal length
Focal length
Dolly zoom

- https://www.youtube.com/watch?v=NB4bikrNzMk
Perspective cues
Perspective cues
Perspective cues
Lens distortion (Fisheye lens)

Multiple models to capture distortion, commonly used is Plumb Bob model

Slide from Jianbo Shi
Lens distortion

- Barrel distortion
- Pincushion distortion
- Moustache distortion

Modeled as a function that changes pixel \((u,v)\) after intrinsics + extrinsics based projection.
Camera Calibration

- Compute camera intrinsic parameters & distortion
- Compute extrinsics between multiple views/different cameras
- **Key idea**: Use a known object of fixed size & match it across multiple scenes -> provides enough constraints to solve for camera parameters
- Good code available online!