ECE276A: Sensing & Estimation in Robotics
Lecture 7: Motion and Observation Models

Instructor:

Nikolay Atanasov: natanasov@ucsd.edu

Teaching Assistants:

Qiaojun Feng: qif007@eng.ucsd.edu

Arash Asgharivaskasi: aasghari@eng.ucsd.edu

Thai Duong: tduong@eng.ucsd.edu

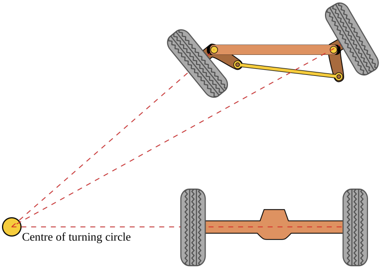
Yiran Xu: y5xu@eng.ucsd.edu

UC San Diego

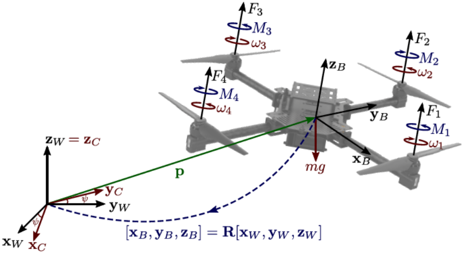
JACOBS SCHOOL OF ENGINEERING

Electrical and Computer Engineering

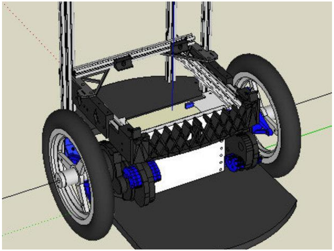
Motion Models



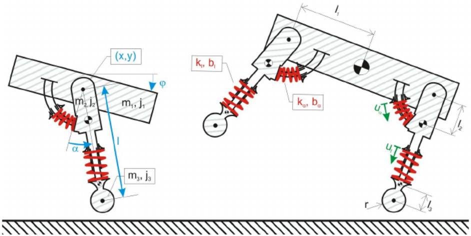
Ackermann Drive



Quadrotor



Differential Drive



Spring-loaded Gait

Motion Model

- A **motion model** is a function $f(\mathbf{x}, \mathbf{u}, \mathbf{w})$ relating the current state $\mathbf{x}$ and control input $\mathbf{u}$ of a robot with its state change subject to motion noise $\mathbf{w}$
  - Continuous-time: $\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t), \mathbf{w}(t))$
  - Discrete-time: $\mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{u}_t, \mathbf{w}_t)$
- Due to the presence of motion noise, the state change $\dot{\mathbf{x}}(t)$ or $\mathbf{x}_{t+1}$ is a random variable and can equivalently be described by its probability density function (pdf) conditioned on $\mathbf{x}$ and $\mathbf{u}$:
  - Continuous-time: $\dot{\mathbf{x}}(t)$ has pdf $p_f(\cdot \mid \mathbf{x}(t), \mathbf{u}(t))$
  - Discrete-time: $\mathbf{x}_{t+1}$ has pdf $p_f(\cdot \mid \mathbf{x}_t, \mathbf{u}_t)$

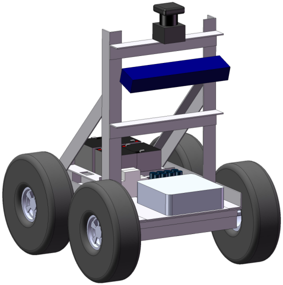
How is a motion model obtained?

- A motion model can be obtained using:
  - Physics-based kinematics or dynamics modeling, e.g., differential-drive model, Ackermann-drive model, quadrotor model, etc.
  - System identification or supervised learning from a dataset $D = \{(\mathbf{x}_i, \mathbf{u}_i, \mathbf{x}_i')\}$ of system transitions
  - Model-based reinforcement learning, where it is inferred indirectly as the robot is learning to perform a task
- **Odometry**, using sensor data (e.g., wheel encoders, IMU, camera, laser) to estimate ego motion in retrospect, after the robot has moved, is an alternative to using a motion model suitable for localization and mapping but not for planning and control.

Differential-drive Kinematic Model

- **State**: $\mathbf{x} = (\mathbf{p}, \theta)$, where $\mathbf{p} = (x, y) \in \mathbb{R}^2$ is position and $\theta \in (-\pi, \pi]$ is orientation (yaw angle)
- **Control**: $\mathbf{u} = (v, \omega)$, where $v \in \mathbb{R}$ is linear velocity and $\omega \in \mathbb{R}$ is angular velocity (yaw rate)
- **Continuous-time model**:

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = f(\mathbf{x}, \mathbf{u}) := \begin{bmatrix} v \cos\theta \\ v \sin\theta \\ \omega \end{bmatrix}$$



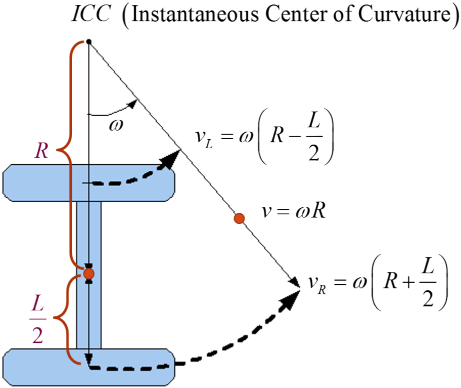
- **Discrete-time model** with time discretization τ:

$$\mathbf{x}_{t+1} = \begin{bmatrix} x_{t+1} \\ y_{t+1} \\ \theta_{t+1} \end{bmatrix} = f(\mathbf{x}_t, \mathbf{u}_t) := \mathbf{x}_t + \tau \begin{bmatrix} v_t \operatorname{sinc}\left(\frac{\omega_t \tau}{2}\right) \cos\left(\theta_t + \frac{\omega_t \tau}{2}\right) \\ v_t \operatorname{sinc}\left(\frac{\omega_t \tau}{2}\right) \sin\left(\theta_t + \frac{\omega_t \tau}{2}\right) \\ \omega_t \end{bmatrix}$$

Continuous-time Differential-drive Kinematic Model

- Let L be the distance between the wheels and R be the radius of rotation, i.e., the distance from the ICC to axel center.
- The arc-length travelled is equal to the angle θ times the radius R

$$\omega = \frac{\theta}{t} \qquad v = \frac{R\theta}{t} = \omega R$$



$$\omega = \frac{v_R - v_L}{L}$$

$$R = \frac{L}{2}\left(\frac{v_L + v_R}{v_R - v_L}\right) = \frac{v}{\omega}$$

$$v = \frac{v_R + v_L}{2}$$

$$\begin{bmatrix}\dot{x}\\ \dot{y}\end{bmatrix} = \begin{bmatrix}\cos\theta & -\sin\theta\\ \sin\theta & \cos\theta\end{bmatrix}\begin{bmatrix}v\\ 0\end{bmatrix}$$

$$\dot{\theta} = \omega$$

Discrete-time Differential-drive Kinematic Model

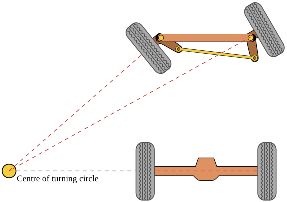
- What is the state after $\tau := t - t_0$ seconds if we apply linear velocity v and angular velocity ω?
- To convert the continuous-time differential-drive model to discrete time, we can solve the ordinary differential equations:

$$
\begin{aligned}
\dot{x}(t) &= v\cos\theta(t) \\
\dot{y}(t) &= v\sin\theta(t) \\
\dot{\theta}(t) &= \omega
\end{aligned}
\quad \Rightarrow \quad
\begin{aligned}
\theta(t) &= \theta(t_0) + \int_{t_0}^{t} \omega ds = \theta(t_0) + \omega(t - t_0) \\
x(t) &= x(t_0) + v\int_{t_0}^{t} \cos\theta(s) ds \\
&= x(t_0) + \frac{v}{\omega}\left(\sin\left(\omega(t - t_0) + \theta(t_0)\right) - \sin\theta(t_0)\right) \\
&= x(t_0) + v(t - t_0)\frac{\sin(\omega(t - t_0)/2)}{\omega(t - t_0)/2}\cos(\theta(t_0) + \omega(t - t_0)/2) \\
y(t) &= y(t_0) + v\int_{t_0}^{t} \sin\theta(s) ds \\
&= y(t_0) - \frac{v}{\omega}\left(\cos\theta(t_0) - \cos\left(\omega(t - t_0) + \theta(t_0)\right)\right) \\
&= y(t_0) + v(t - t_0)\frac{\sin(\omega(t - t_0)/2)}{\omega(t - t_0)/2}\sin(\theta(t_0) + \omega(t - t_0)/2)
\end{aligned}
$$

Ackermann-drive Kinematic Model

- **State**: $\mathbf{x} = (\mathbf{p}, \theta)$, where $\mathbf{p} = (x, y) \in \mathbb{R}^2$ is position and $\theta \in (-\pi, \pi]$ is orientation (yaw angle)
- **Control**: $\mathbf{u} = (v, \phi)$, where $v \in \mathbb{R}$ is linear velocity and $\phi \in (-\pi, \pi]$ is steering angle
- **Continuous-time model**:

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = f(\mathbf{x}, \mathbf{u}) := \begin{bmatrix} v \cos\theta \\ v \sin\theta \\ \frac{v}{L} \tan\phi \end{bmatrix}$$



where L is the distance between the wheels

- With the definition $\omega := \frac{v}{L} \tan\phi$, the model is equivalent to the differential-drive model

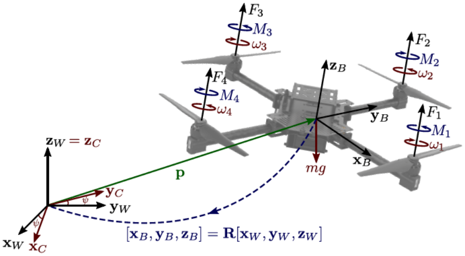
- **Discrete-time model** with time discretization τ and $\omega_t := \frac{v_t}{L} \tan\phi_t$:

$$\mathbf{x}_{t+1} = \begin{bmatrix} x_{t+1} \\ y_{t+1} \\ \theta_{t+1} \end{bmatrix} = f(\mathbf{x}_t, \mathbf{u}_t) := \mathbf{x}_t + \tau \begin{bmatrix} v_t \text{sinc}\left(\frac{\omega_t \tau}{2}\right) \cos\left(\theta_t + \frac{\omega_t \tau}{2}\right) \\ v_t \text{sinc}\left(\frac{\omega_t \tau}{2}\right) \sin\left(\theta_t + \frac{\omega_t \tau}{2}\right) \\ \omega_t \end{bmatrix}$$

Quadrotor Dynamics Model

- **State**: $\mathbf{x} = (\mathbf{p}, \dot{\mathbf{p}}, R, \boldsymbol{\omega})$ with position $\mathbf{p} \in \mathbb{R}^3$, velocity $\dot{\mathbf{p}} \in \mathbb{R}^3$, orientation $R \in SO(3)$, and body-frame rotational velocity $\boldsymbol{\omega} \in \mathbb{R}^3$
- **Control**: $\mathbf{u} = (\rho, \tau)$ with thrust force $\rho \in \mathbb{R}$ and torque $\tau \in \mathbb{R}^3$
- **Continuous-time model** with mass $m \in \mathbb{R}_{>0}$, gravitational acceleration g, moment of inertia $J \in \mathbb{R}^{3\times3}$ and z-axis $\mathbf{e}_3 \in \mathbb{R}^3$:

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}) = \begin{cases} m\ddot{\mathbf{p}} = -mg\mathbf{e}_3 + \rho R\mathbf{e}_3 \\ \dot{R} = R\left[\boldsymbol{\omega}\right]_\times \\ J\dot{\boldsymbol{\omega}} = -\boldsymbol{\omega} \times J\boldsymbol{\omega} + \tau \end{cases}$$



Odometry-based Motion Model

- Onboard sensors (camera, lidar, encoders, imu, etc.) may be used to estimate the relative transformation:

$$ {}_t\hat{T}_{t+1} := \begin{bmatrix} {}_t\hat{R}_{t+1} & {}_t\hat{\mathbf{p}}_{t+1} \\ \mathbf{0}^\top & 1 \end{bmatrix} \in SE(3) $$

  of the robot pose at time $t+1$ with respect to the body frame at time t

- Assuming a small time discretization, the estimates ${}_t\hat{T}_{t+1}$ are accurate

- Let $\mathbf{x}_0 := {}_W T_0$ be a known initial robot pose. To simplify notation, we will drop the $\{W\}$ subscript when referring to the world frame.

- An odometry motion model estimates the robot pose $\mathbf{x}_{t+1}$ at time $t+1$ (specifying the transformation from the body frame at time $t+1$ to the world frame) using the relative pose estimates ${}_0\hat{T}_1, \ldots, {}_t\hat{T}_{t+1}$

Odometry-based Motion Model

- Given $\left\{\mathbf{u}_\tau := {}_\tau\hat{T}_{\tau+1} \mid \tau = 0, \ldots, t\right\}$ and using the composition rule of transformations, we can estimate the robot pose at time $t+1$:

$$\begin{aligned}\mathbf{x}_{t+1} = T_{t+1} \approx T_t \;\; {}_t\hat{T}_{t+1} = \mathbf{x}_t \oplus \mathbf{u}_t =: f(\mathbf{x}_t, \mathbf{u}_t)\\ \approx \mathbf{x}_{t-1} \oplus \mathbf{u}_{t-1} \oplus \mathbf{u}_t \approx \ldots \approx \mathbf{x}_0 \oplus_{\tau=0}^{t} \mathbf{u}_\tau = T_0 \prod_{\tau=0}^{t} {}_\tau\hat{T}_{\tau+1}\end{aligned}$$

- The operation $\oplus$ denotes composition of $SE(3)$ elements

- The odometry estimate is "drifting", i.e., gets worse and worse over time, because the small errors in each ${}_t\hat{T}_{t+1}$ are accumulated

Observation Models

Inertial Measurement Unit

RGB Camera

Global Positioning System

2-D Lidar

Observation Model

- An **observation model** is a function $\mathbf{z} = h(\mathbf{x}, \mathbf{m}, \mathbf{v})$ relating the robot state $\mathbf{x}$ and the surrounding environment $\mathbf{m}$ with the sensor observation $\mathbf{z}$ subject to measurement noise $\mathbf{v}$:

$$\mathbf{z}_t = h(\mathbf{x}_t, \mathbf{m}_t, \mathbf{v}_t)$$

- Due to the presence of measurement noise, the observation $\mathbf{z}_t$ is a random variable and can equivalently be described by its pdf conditioned on $\mathbf{x}_t$ and $\mathbf{m}_t$:

$$\mathbf{z}_t \quad \text{has pdf} \quad p_h(\cdot \mid \mathbf{x}_t, \mathbf{m}_t)$$

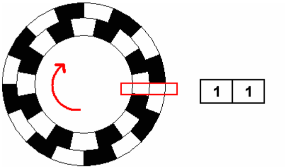
- Common sensor models:
  - **Inertial**: encoders, magnetometer, gyroscope, accelerometer
  - **Position model**: direct position measurements, e.g., GPS, RGBD camera, laser scanner
  - **Bearing model**: angular measurements to points in 3-D, e.g., compass, RGB camera
  - **Range model**: distance measurements to points in 3-D, e.g., radio received signal strength (RSS) or time-of-flight

Encoders

- A magnetic encoder consists of a rotating gear, a permanent magnet, and a sensing element
- The sensor has two output channels with offset phase to determine the direction of rotation
- A microcontroller counts the number of transitions adding or subtracting 1 (depending on the direction of rotation) to the counter

- The distance traveled by the wheel, corresponding to one tick on the encoder is:

$$\text{meters per tick} = \frac{\pi \times (\text{wheel diameter})}{\text{ticks per revolution}}$$



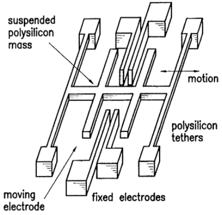
- The distance traveled during time τ for a given encoder count z, wheel diameter d, and 360 ticks per revolution is:

$$\tau v \approx \frac{\pi d z}{360}$$

and can be used to predict position change in a differential-drive model

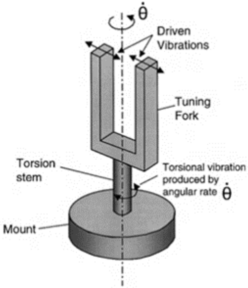
MEMS Strapdown IMU

- **MEMS**: micro-electro-mechanical system
- **IMU**: inertial measurement unit:
  - triaxial accelerometer
  - triaxial gyroscope (measures angular velocity)
  - **Strapdown**: the IMU and the object/vehicle inertial frames are joined together/identical



Surface Micromachined Accelerometer

- **Accelerometer**:
  - A mass m on a spring with constant k. The spring displacement is prop. to the system acceleration: $F = ma = kd \quad \Rightarrow \quad d = \frac{ma}{k}$
  - VLSI Fabrication: the displacement of a metal plate with mass m is measured with respect to another plate using capacitance
  - Used for car airbags (if the acceleration goes above $2g$, the car is hitting something!)



- **Gyroscope**: uses Coriolis force to detect rotational velocity from the changing mechanical resonsance of a tuning fork

IMU Observation Model

- **State**: $(\mathbf{p}, \dot{\mathbf{p}}, \ddot{\mathbf{p}}, R, \boldsymbol{\omega}, \dot{\boldsymbol{\omega}}, \mathbf{b}_g, \mathbf{b}_a)$ with position $\mathbf{p} \in \mathbb{R}^3$, velocity $\dot{\mathbf{p}} \in \mathbb{R}^3$, acceleration $\ddot{\mathbf{p}} \in \mathbb{R}^3$, orientation $R \in SO(3)$, rotational velocity $\boldsymbol{\omega} \in \mathbb{R}^3$ (body frame), and rotational acceleration $\dot{\boldsymbol{\omega}} \in \mathbb{R}^3$ (body frame), gyroscope bias $\mathbf{b}_g \in \mathbb{R}^3$, accelerometer bias $\mathbf{b}_a \in \mathbb{R}^3$
- **Extrinsic Parameters**: the IMU position ${}_B\mathbf{p}_I \in \mathbb{R}^3$ and orientation ${}_BR_I \in SO(3)$ in the body frame (assumed known or obtained via calibration)
- **Measurement**: $(\mathbf{z}_\omega, \mathbf{z}_a)$ with rotational velocity measurement $\mathbf{z}_\omega \in \mathbb{R}^3$ and linear acceleration measurement $\mathbf{z}_a \in \mathbb{R}^3$

IMU Observation Model

- **Continuous-time model**: with gravitational acceleration g, gyro measurement noise $\mathbf{n}_g \in \mathbb{R}^3$, accelerometer measurement noise $\mathbf{n}_a \in \mathbb{R}^3$ (assumed zero-mean white Gaussian):

$$\begin{aligned}
\mathbf{z}_\omega &= {}_B R_I^\top \boldsymbol{\omega} + \mathbf{b}_g + \mathbf{n}_g \\
\mathbf{z}_a &= {}_W R_I^\top \left({}_W \ddot{\mathbf{p}}_I - g\mathbf{e}_3\right) + \mathbf{b}_a + \mathbf{n}_a \\
&= \left(R\ {}_B R_I\right)^\top \left(\frac{d}{dt^2}\left(\mathbf{p} + R\ {}_B\mathbf{p}_I\right) - g\mathbf{e}_3\right) + \mathbf{b}_a + \mathbf{n}_a \\
&= {}_B R_I^\top \left(R^\top(\ddot{\mathbf{p}} - g\mathbf{e}_3) + [\dot{\boldsymbol{\omega}}]_\times\ {}_B\mathbf{p}_I + [\boldsymbol{\omega}]_\times^2\ {}_B\mathbf{p}_I\right) + \mathbf{b}_a + \mathbf{n}_a
\end{aligned}$$

- For a strapdown IMU (${}_B R_I = I$ and ${}_B\mathbf{p}_I = \mathbf{0}$), the above simplifies to:

$$\begin{aligned}
\mathbf{z}_\omega &= \boldsymbol{\omega} + \mathbf{b}_g + \mathbf{n}_g \\
\mathbf{z}_a &= R^\top(\ddot{\mathbf{p}} - g\mathbf{e}_3) + \mathbf{b}_a + \mathbf{n}_a
\end{aligned}$$

- **Discrete-time model**: A. Mourikis and S. Roumeliotis, "A Multi-State Constraint Kalman Filter for Vision-aided Inertial Navigation"

Lasers

Single-beam Garmin Lidar

2-D Hokuyo Lidar

3-D Velodyne Lidar

LIDAR Model

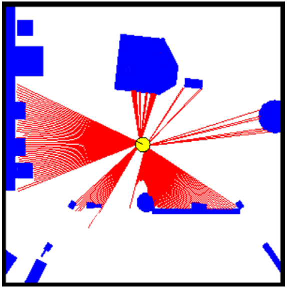
- **Lidar**: LIght Detection And Ranging

- Illuminates the scene with pulsed laser light and measures the return times and wavelengths of the reflected pulses

- Mirrors are used to steer the laser beam in the *xy* plane (and *zy* plane for 3D lidars)

- LIDAR rays are emitted over a set of known horizontal (azimuth) and vertical (elevation) angles $\{\alpha_k, \epsilon_k\}$ and return range estimates $\{r_k\}$ to obstacles in the environment $\mathbf{m}$

- Example: Hokuyo URG-04LX; detectable range: 0.02 to 4m; 240° field of view with 0.36° angular resolution (666 beams); 100 ms/scan



Laser Range-Azimuth-Elevation Model

- Consider a Lidar with position $\mathbf{p} \in \mathbb{R}^3$ and orientation $R \in SO(3)$ observing a point $\mathbf{m} \in \mathbb{R}^3$ in the world frame
- The point $\mathbf{m}$ has coordinates $\bar{\mathbf{m}} := R^\top(\mathbf{m} - \mathbf{p})$ in the lidar frame
- The lidar provides a spherical coordinate measurement of $\bar{\mathbf{m}}$:

$$\bar{\mathbf{m}} = R^\top(\mathbf{m} - \mathbf{p}) = \begin{bmatrix} r \cos\alpha \cos\epsilon \\ r \sin\alpha \cos\epsilon \\ r \sin\epsilon \end{bmatrix}$$

  where r is the range, α is the azimuth, and ϵ is the elevation

- **Inverse observation model**: expresses the lidar state $\mathbf{p}$, R and environment state $\mathbf{m}$, in terms of the measurement $\mathbf{z} = \begin{bmatrix} r & \alpha & \epsilon \end{bmatrix}^T$
- Inverting gives the **laser range-azimuth-elevation model**:

$$\mathbf{z} = \begin{bmatrix} r \\ \alpha \\ \epsilon \end{bmatrix} = \begin{bmatrix} \|\bar{\mathbf{m}}\|_2 \\ \arctan\left(\bar{\mathbf{m}}_y / \bar{\mathbf{m}}_x\right) \\ \arcsin\left(\bar{\mathbf{m}}_z / \|\bar{\mathbf{m}}\|_2\right) \end{bmatrix} \qquad \bar{\mathbf{m}} = R^\top(\mathbf{m} - \mathbf{p})$$

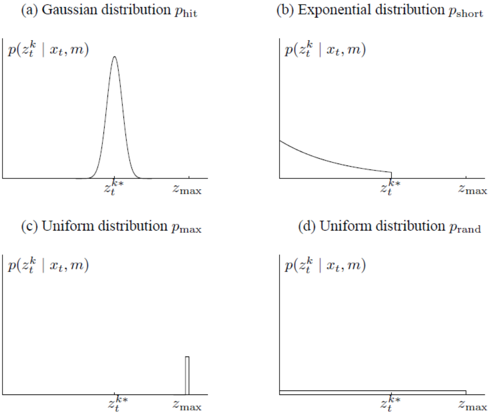
Laser Beam Model

- Let r_t^k be the range measurement of beam k from pose $\mathbf{x}_t$ in map $\mathbf{m}$
- Let r_t^{k*} be the expected measurement and let r_{max} be the max range
- The laser beam model assumes that the **beams are independent**:

$$p_h(r_t \mid \mathbf{x}_t, \mathbf{m}) = \prod_k p(r_t^k \mid \mathbf{x}_t, \mathbf{m})$$

Four types of measurement noise:

1. Small measurement noise: p_{hit}, Gaussian
2. Unexpected object: p_{short}, Exponential
3. Unexplained noise: p_{rand}, Uniform
4. No objects hit: p_{max}, Uniform



Laser Beam Model

- Independent beam assumption: $p_h(r_t \mid \mathbf{x}_t, \mathbf{m}) = \prod_k p(r_t^k \mid \mathbf{x}_t, \mathbf{m})$
- Each beam likelihood is a mixture model of four noise types:

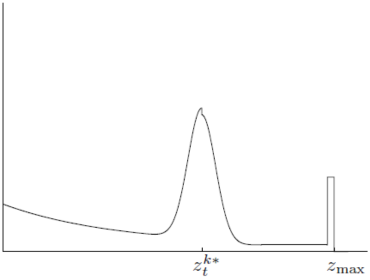
$$p(r_t^k \mid \mathbf{x}_t, \mathbf{m}) = \alpha_1 p_{hit}(r_t^k \mid \mathbf{x}_t, \mathbf{m}) + \alpha_2 p_{short}(r_t^k \mid \mathbf{x}_t, \mathbf{m}) + \alpha_3 p_{rand}(r_t^k \mid \mathbf{x}_t, \mathbf{m}) + \alpha_4 p_{max}(r_t^k \mid \mathbf{x}_t, \mathbf{m})$$

$$p_{hit}(r_t^k \mid \mathbf{x}, \mathbf{m}) = \begin{cases} \frac{\phi(r_t^k; r_t^{k*}, \sigma^2)}{\int_0^{r_{max}} \phi(s; r_t^{k*}, \sigma^2) ds} & \text{if } 0 \le r_t^k \le r_{max} \\ 0 & \text{else} \end{cases}$$

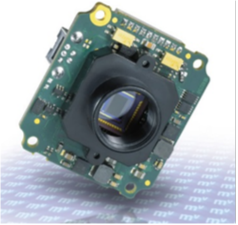
$$p_{short}(r_t^k \mid \mathbf{x}, \mathbf{m}) = \begin{cases} \frac{\lambda_s e^{-\lambda_s r_t^{k*}}}{1 - e^{-\lambda_s r_t^{k*}}} & \text{if } 0 \le r_t^k \le r_t^{k*} \\ 0 & \text{else} \end{cases}$$

$$p_{rand}(r_t^k \mid \mathbf{x}, \mathbf{m}) = \begin{cases} \frac{1}{r_{max}} & \text{if } 0 \le r_t^k < r_{max} \\ 0 & \text{else} \end{cases}$$

$$p_{max}(r_t^k \mid \mathbf{x}, \mathbf{m}) = \delta(r_t^k; r_{max}) := \begin{cases} 1 & \text{if } r_t^k = r_{max} \\ 0 & \text{else} \end{cases}$$



Cameras



Global shutter

Stereo (+ IMU)

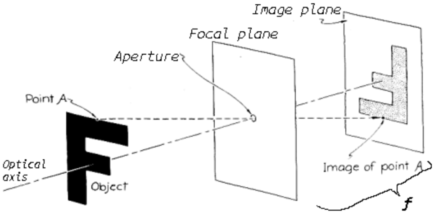
RGBD

Event-based

Image Formation

- **Image formation model**: must trade-off physical constraints and mathematical simplicity
- The values of an image depend on the shape and reflectance of the scene as well as the distribution of light
- **Image intensity/brightness/irradiance** $I(u, v)$ describes the energy falling onto a small patch of the imaging sensor (integrated both over the shutter interval and over a region of space) and is measured in power per unit area (W/m^2)
- A camera uses a set of lenses to control the direction of light propagation by means of *diffraction*, *refraction*, and *reflection*
- **Thin lens model**: a simple geometric model of image formation that considers only refraction
- **Pinhole model**: a thin lens model in which the lens aperture is decreased to zero and all rays are forced to go through the optical center and remain undeflected (diffraction becomes dominant).

Pinhole Camera Model



- **Focal plane**: perpendicular to the **optical axis** with a circular aperture at the **optical center**

- **Image plane**: parallel to the focal plane and a distance f (**focal length**) in **meters** from the optical center

- The pinhole camera model is described in an **optical frame** centered at the optical center with the optical axis as the z-axis:
  - optical frame: x = right, y = down, z = forward
  - world frame: x = forward, y = left, z = up

- **Image flip**: the object appears upside down on the image plane. To eliminate this effect, we can simply flip the image $(x, y) \to (-x, -y)$, which corresponds to placing the image plane $\{z = -f\}$ in front of the optical center instead of behind $\{z = f\}$.

Pinhole Camera Model

- **Field of view**: the angle subtended by the spatial extend of the image plane as seen from the optical center. If s is the side of the image plane in **meters**, then the field of view is $\boxed{\theta = 2\arctan\left(\frac{s}{2f}\right)}$.
  - For a flat image plane: $\theta < 180^\circ$.
  - For a spherical or ellipsoidal imaging surface, common in omnidirectional cameras, θ can exceed 180°.

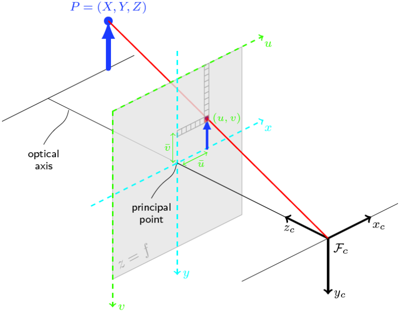
- **Ray tracing**: under assumptions of the pinhole model and Lambertian surfaces, image formation can be reduced to tracing rays from points on objects to pixels. A mathematical model associating 3-D points in the world frame to 2-D points in the image frame must account for:
  1. **Extrinsics**: world-to-camera frame transformation
  2. **Projection**: 3D-to-2D coordinate projection
  3. **Intrinsics**: scaling and translation of the image coordinate frame

Extrinsics

- Let $\mathbf{p} \in \mathbb{R}^3$ and $R \in SO(3)$ be the camera position and orientation in the world frame
- Rotation from a regular to an optical frame: ${}_oR_r := \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & -1 \\ 1 & 0 & 0 \end{bmatrix}$
- Let (X_w, Y_w, Z_w) be the coordinates of point $\mathbf{m}$ in the world frame. The coordinates of $\mathbf{m}$ in the optical frame are then:

$$\begin{pmatrix} X_o \\ Y_o \\ Z_o \\ 1 \end{pmatrix} = \begin{bmatrix} {}_oR_r & \mathbf{0} \\ \mathbf{0}^\top & 1 \end{bmatrix} \begin{bmatrix} R & \mathbf{p} \\ \mathbf{0}^\top & 1 \end{bmatrix}^{-1} \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix} = \begin{bmatrix} {}_oR_r R^\top & -{}_oR_r R^\top \mathbf{p} \\ 0 & 1 \end{bmatrix} \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix}$$

Projection



- The 3D-to-2D perspective projection operation relates the optical-frame coordinates (X_o, Y_o, Z_o) of point $\mathbf{m}$ to its image coordinates (x, y) using similar triangles:

$$x = f\frac{X_o}{Z_o} \qquad y = f\frac{Y_o}{Z_o}$$

$$\begin{pmatrix} z \\ y \\ 1 \end{pmatrix} = \frac{1}{Z_o}\begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}\begin{pmatrix} X_o \\ Y_o \\ Z_o \\ 1 \end{pmatrix}$$

- The above can be decomposed into:

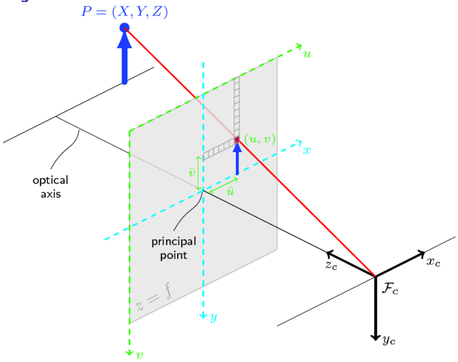
$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \underbrace{\begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{image flip: } F_f}\underbrace{\begin{bmatrix} -f & 0 & 0 \\ 0 & -f & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{focal scaling: } K_f}\underbrace{\frac{1}{Z_o}\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\text{canonical projection: } \pi}\begin{pmatrix} X_o \\ Y_o \\ Z_o \\ 1 \end{pmatrix}$$

Intrinsics

- Images are obtained in terms of pixels (u, v) with the origin of the pixel array typically in the upper-left corner of the image.
- The relationship between the image frame and the pixel array is specified via the following parameters:
  - (s_u, s_v) [pixels/meter]: define the **scaling** from meters to pixels and the **aspect ration** $\sigma = s_u/s_v$
  - (c_u, c_v) [pixels]: coordinates of the *principal point* used to translate the image frame origin, e.g., $(c_u, c_v) = (320.5, 240.5)$ for a 640×480 image
  - s_θ [pixels/meter]: **skew factor** that scales non-rectangular pixels and is proportional to $\cot(\alpha)$ where α is the angle between the coordinate axes of the pixel array.
- Normalized coordinates in the image frame are converted to pixel coordinates in the pixel array using the **intrinsic parameter matrix**:

$$\underbrace{\begin{bmatrix} s_u & s_\theta & c_u \\ 0 & s_v & c_v \\ 0 & 0 & 1 \end{bmatrix}}_{\text{pixel scaling: } K_s} \underbrace{\begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{image flip: } F_f} \underbrace{\begin{bmatrix} -f & 0 & 0 \\ 0 & -f & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{focal scaling: } K_f} = \underbrace{\begin{bmatrix} fs_u & fs_\theta & c_u \\ 0 & fs_v & c_v \\ 0 & 0 & 1 \end{bmatrix}}_{\text{calibration matrix: } K} \in \mathbb{R}^{3\times 3}$$

Pinhole Camera Model Summary



- **Extrinsics**:

$$\begin{pmatrix} X_o \\ Y_o \\ Z_o \\ 1 \end{pmatrix} = \begin{bmatrix} {}_oR_r R^\top & -{}_oR_r R^\top \mathbf{p} \\ \mathbf{0}^\top & 1 \end{bmatrix} \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix}$$

- **Projection and Intrinsics**:

$$\underbrace{\begin{pmatrix} u \\ v \\ 1 \end{pmatrix}}_{\text{pixels}} = \underbrace{\begin{bmatrix} fs_u & fs_\theta & c_u \\ 0 & fs_v & c_v \\ 0 & 0 & 1 \end{bmatrix}}_{\text{calibration: } K} \underbrace{\frac{1}{Z_o} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\text{canonical projection: } \pi} \begin{pmatrix} X_o \\ Y_o \\ Z_o \\ 1 \end{pmatrix}$$

Projective Camera Models

- The **canonical projection function** for vector $\mathbf{x} \in \mathbb{R}^3$ is:

$$\pi(\mathbf{x}) := \frac{1}{\mathbf{e}_3^\top \mathbf{x}} \mathbf{x}$$

- **Perspective projection model**: the pixel coordinates $\mathbf{z} \in \mathbb{R}^2$ of a point $\mathbf{m} \in \mathbb{R}^3$ in the world frame observed by a camera at position $\mathbf{p} \in \mathbb{R}^3$ with orientation $R \in SO(3)$ and intrinsic parameters $K \in \mathbb{R}^{3\times3}$ are:

$$\underline{\mathbf{z}} = PK\pi({}_oR_r R^\top(\mathbf{m} - \mathbf{p})) \qquad\qquad P := \begin{bmatrix} I & \mathbf{0} \end{bmatrix} \in \mathbb{R}^{2\times3}$$

- **Spherical perspective projection**: if the imaging surface is a sphere $\mathbb{S}^2 := \{\mathbf{x} \in \mathbb{R}^3 \mid \|\mathbf{x}\| = 1\}$ (motivated by retina shapes in biological systems), we can define a spherical projection $\pi_s(\mathbf{x}) = \frac{\mathbf{x}}{\|\mathbf{x}\|_2}$ and use it in place of π in the model above.

- **Catadioptric model**: uses an ellipsoidal imaging surface

Radial distortion

- **Wide field of view camera**: in addition to linear distortions described by the intrinsic parameters K, one can observe distortion along radial directions.

- The simplest effective **model for radial distortion**:

$$x = x_d(1 + a_1 r^2 + a_2 r^4)$$
$$y = y_d(1 + a_1 r^2 + a_2 r^4)$$

  where (x_d, y_d) are the pixel coordinates of distorted points and $r^2 = x_d^2 + y_d^2$ and a_1, a_2 are additional parameters modeling the amount of distortion.

Epipolar Geometry

- Let $\mathbf{m} \in \mathbb{R}^3$ (world frame) be observed by two **calibrated** cameras
- Without loss of generality assume that the first camera frame coincides with the world frame. Let the position and orientation of the second camera be $\mathbf{p} \in \mathbb{R}^3$ and $R \in SO(3)$ (absorb ${}_oR_r$ into R)
- Let $\underline{\mathbf{z}}_1$, $\underline{\mathbf{z}}_2$ be the homogeneous pixel coordinates of $\mathbf{m}$ in the two images
- Let $\underline{\mathbf{y}}_i := K^{-1}\underline{\mathbf{z}}_i$ be the normalized pixel coordinates so that:

$$\begin{aligned} \lambda_1 \underline{\mathbf{y}}_1 &= \mathbf{m}, & \lambda_1 &= \mathbf{e}_3^\top \mathbf{m} = \text{unknown scale} \\ \lambda_2 \underline{\mathbf{y}}_2 &= R^\top(\mathbf{m} - \mathbf{p}), & \lambda_2 &= \mathbf{e}_3^\top R^\top(\mathbf{m} - \mathbf{p}) = \text{unknown scale} \end{aligned}$$

- We obtain the following relationship between the image points:

$$\lambda_1 \underline{\mathbf{y}}_1 = R\lambda_2 \underline{\mathbf{y}}_2 + \mathbf{p}$$

- To eliminate the unknown depths λ_i, pre-multiply by $[\mathbf{p}]_\times$ and note that $[\mathbf{p}]_\times \underline{\mathbf{y}}_1$ is perpendicular to $\underline{\mathbf{y}}_1$:

$$\underbrace{\lambda_1 \underline{\mathbf{y}}_1^\top [\mathbf{p}]_\times \underline{\mathbf{y}}_1}_{0} = \lambda_2 \underline{\mathbf{y}}_1^\top [\mathbf{p}]_\times R \underline{\mathbf{y}}_2 + \underbrace{\underline{\mathbf{y}}_1^\top [\mathbf{p}]_\times \mathbf{p}}_{0}$$

Essential Matrix

- Thus, $\lambda_2 \underline{\mathbf{y}}_1^\top [\mathbf{p}]_\times R \underline{\mathbf{y}}_2 = 0$ and since $\lambda_2 > 0$, we arrive at the following
- **Epipolar constraint**: Consider observations $\underline{\mathbf{y}}_1 = K_1^{-1}\underline{\mathbf{z}}_1$, $\underline{\mathbf{y}}_2 = K_2^{-1}\underline{\mathbf{z}}_2$ in normalized image coordinates of the same point $\mathbf{m}$ from two calibrated cameras with relative pose $(R, \mathbf{p})$ of camera 2 in the frame of camera 1. Then:
$$0 = \underline{\mathbf{y}}_1^\top \left([\mathbf{p}]_\times R\right) \underline{\mathbf{y}}_2 = \underline{\mathbf{y}}_1^\top E \underline{\mathbf{y}}_2$$
where $E := [\mathbf{p}]_\times R \in \mathbb{R}^{3\times 3}$ is the **essential matrix**.
- **Essential matrix characterization**: a non-zero $E \in \mathbb{R}^{3\times 3}$ is an essential matrix iff its singular value decomposition is $E = U\mathbf{diag}(\sigma, \sigma, 0)V^T$ for some $\sigma \geq 0$ and $U, V \in SO(3)$
- **Pose recovery from the Essential matrix**: There are exactly two relative poses corresponding to a non-zero essential matrix E:
$$\left([\mathbf{p}]_\times, R\right) = \left(UR_z\left(\frac{\pi}{2}\right)\mathbf{diag}(\sigma, \sigma, 0)U^T, UR_z^T\left(\frac{\pi}{2}\right)V^T\right)$$
$$\left([\mathbf{p}]_\times, R\right) = \left(UR_z\left(-\frac{\pi}{2}\right)\mathbf{diag}(\sigma, \sigma, 0)U^T, UR_z^T\left(-\frac{\pi}{2}\right)V^T\right)$$

Fundamental Matrix

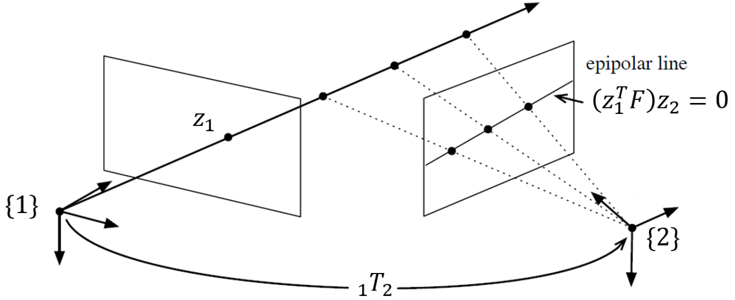
- The epipolar constraint holds even for two **uncalibrated** cameras
- Consider images $\underline{\mathbf{z}}_1 = K_1\underline{\mathbf{y}}_1$ and $\underline{\mathbf{z}}_2 = K_2\underline{\mathbf{y}}_2$ of the same point $\mathbf{m} \in \mathbb{R}^3$ from two uncalibrated cameras with intrinsic parameter matrices K_1 and K_2 and relative pose $(R, \mathbf{p})$ of camera 2 in the frame of camera 1:

$$0 = \underline{\mathbf{y}}_1^\top [\mathbf{p}]_\times R\underline{\mathbf{y}}_2 = \underline{\mathbf{y}}_1^\top E\underline{\mathbf{y}}_2 = \underline{\mathbf{z}}_1^\top K_1^{-\top} E K_2^{-1}\underline{\mathbf{z}}_2 = \underline{\mathbf{z}}_1^\top F\underline{\mathbf{z}}_2$$

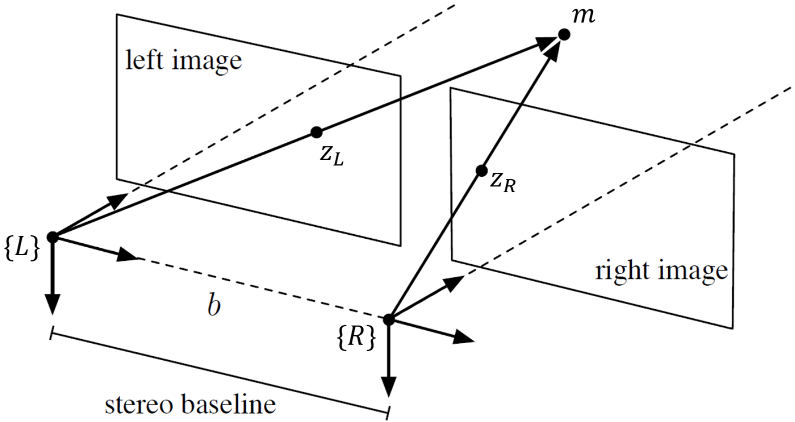
- The matrix $F := K_1^{-T} [\mathbf{p}]_\times R K_2^{-1}$ is called the **fundamental matrix**

Epipolar Line

- If a point $\mathbf{m} \in \mathbb{R}^3$ is observed as $\mathbf{z}_1$ in one image and the fundamental matrix F between two camera frames is known, the epipolar constraint desribes an **epipolar line**, along which the observation $\mathbf{z}_2$ of $\mathbf{m}$ must lie
- The epipolar line is used to limit the search for matching points
- This is possible because the camera model is an affine transformation, i.e., a straight line in Euclidean space, projects to a straight line in image space



Stereo Camera Model



Stereo Camera Model

- **Stereo Camera**: two perspective cameras rigidly connected to one another with a known transformation
- Unlike a single camera, a stereo camera can determine the depth of a point from a single stereo observation
- **Stereo Baseline**: the transformation between the two stereo cameras is only a displacement along the x-axis (optical frame) of size b
- The pixel coordinates $\mathbf{z}_L, \mathbf{z}_R \in \mathbb{R}^2$ of a point $\mathbf{m} \in \mathbb{R}^3$ in the world frame observed by a stereo camera at position $\mathbf{p} \in \mathbb{R}^3$ and orientation $R \in SO(3)$ with intrinsic parameters $K \in \mathbb{R}^{3\times 3}$ are:

$$\underline{\mathbf{z}}_L = K\pi\left({}_oR_r R^\top(\mathbf{m}-\mathbf{p})\right) \qquad \underline{\mathbf{z}}_R = K\pi\left({}_oR_r R^\top(\mathbf{m}-\mathbf{p}) - b\mathbf{e}_1\right)$$

Stereo Camera Model

- Stacking the two observations together gives the stereo camera model:

$$\begin{bmatrix} u_L \\ v_L \\ u_R \\ v_R \end{bmatrix} = \underbrace{\begin{bmatrix} fs_u & 0 & c_u & 0 \\ 0 & fs_v & c_v & 0 \\ fs_u & 0 & c_u & -fs_u b \\ 0 & fs_v & c_v & 0 \end{bmatrix}}_{M} \frac{1}{z} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \qquad \begin{bmatrix} x \\ y \\ z \end{bmatrix} = {}_oR_r R^\top (\mathbf{m} - \mathbf{p})$$

- Because of the stereo steup, two rows of M are identical. The vertical coordinates of the two pixel observations are always the same because the epipolar lines in the stereo configuation are horizontal.

- The v_R equation may be dropped, while the u_R equation is replaced with a **disparity** measurement $\boxed{d = u_L - u_R = \frac{1}{z} fs_u b}$ leading to:

$$\begin{bmatrix} u_L \\ v_L \\ d \end{bmatrix} = \begin{bmatrix} fs_u & 0 & c_u & 0 \\ 0 & fs_v & c_v & 0 \\ 0 & 0 & 0 & fs_u b \end{bmatrix} \frac{1}{z} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \qquad \begin{bmatrix} x \\ y \\ z \end{bmatrix} = {}_oR_r R^\top (\mathbf{m} - \mathbf{p})$$

Observation Models Summary

- **Position sensor**: state $\mathbf{x} = (\mathbf{p}, R)$, position $\mathbf{p} \in \mathbb{R}^3$, orientation $R \in SO(3)$, observed point $\mathbf{m} \in \mathbb{R}^3$, measurement $\mathbf{z} \in \mathbb{R}^3$:
$$\mathbf{z} = h(\mathbf{x}, \mathbf{m}) = R^\top(\mathbf{m} - \mathbf{p})$$
- **Range sensor**: state $\mathbf{x} = (\mathbf{p}, R)$, position $\mathbf{p} \in \mathbb{R}^3$, orientation $R \in SO(3)$, observed point $\mathbf{m} \in \mathbb{R}^3$, measurement $z \in \mathbb{R}$:
$$z = h(\mathbf{x}, \mathbf{m}) = \|R^\top(\mathbf{m} - \mathbf{p})\|_2 = \|\mathbf{m} - \mathbf{p}\|_2$$
- **Bearing sensor**: state $\mathbf{x} = (\mathbf{p}, \theta)$, position $\mathbf{p} \in \mathbb{R}^2$, orientation $\theta \in (-\pi, \pi]$, observed point $\mathbf{m} \in \mathbb{R}^2$, bearing $z \in (-\pi, \pi]$:
$$z = h(\mathbf{x}, \mathbf{m}) = \arctan\left(\frac{m_2 - p_2}{m_1 - p_1}\right) - \theta$$
- **Camera sensor**: state $\mathbf{x} = (\mathbf{p}, R)$, position $\mathbf{p} \in \mathbb{R}^3$, orientation $R \in SO(3)$, intrinsic camera matrix $K \in \mathbb{R}^{3\times 3}$, projection matrix $P := [I,\ \mathbf{0}] \in \mathbb{R}^{2\times 3}$, observed point $\mathbf{m} \in \mathbb{R}^3$, pixel $\mathbf{z} \in \mathbb{R}^2$:
$$\mathbf{z} = h(\mathbf{x}, \mathbf{m}) = PK\pi(R^\top(\mathbf{m} - \mathbf{p})) \qquad \text{projection:} \quad \pi(\mathbf{m}) := \frac{1}{\mathbf{e}_3^\top \mathbf{m}}\mathbf{m}$$