

Illumination Invariant Image Processing for Automated Driving Functions

Automotive Sensors and Actuators

4/16/2024

Daniel Schneider

Motivation

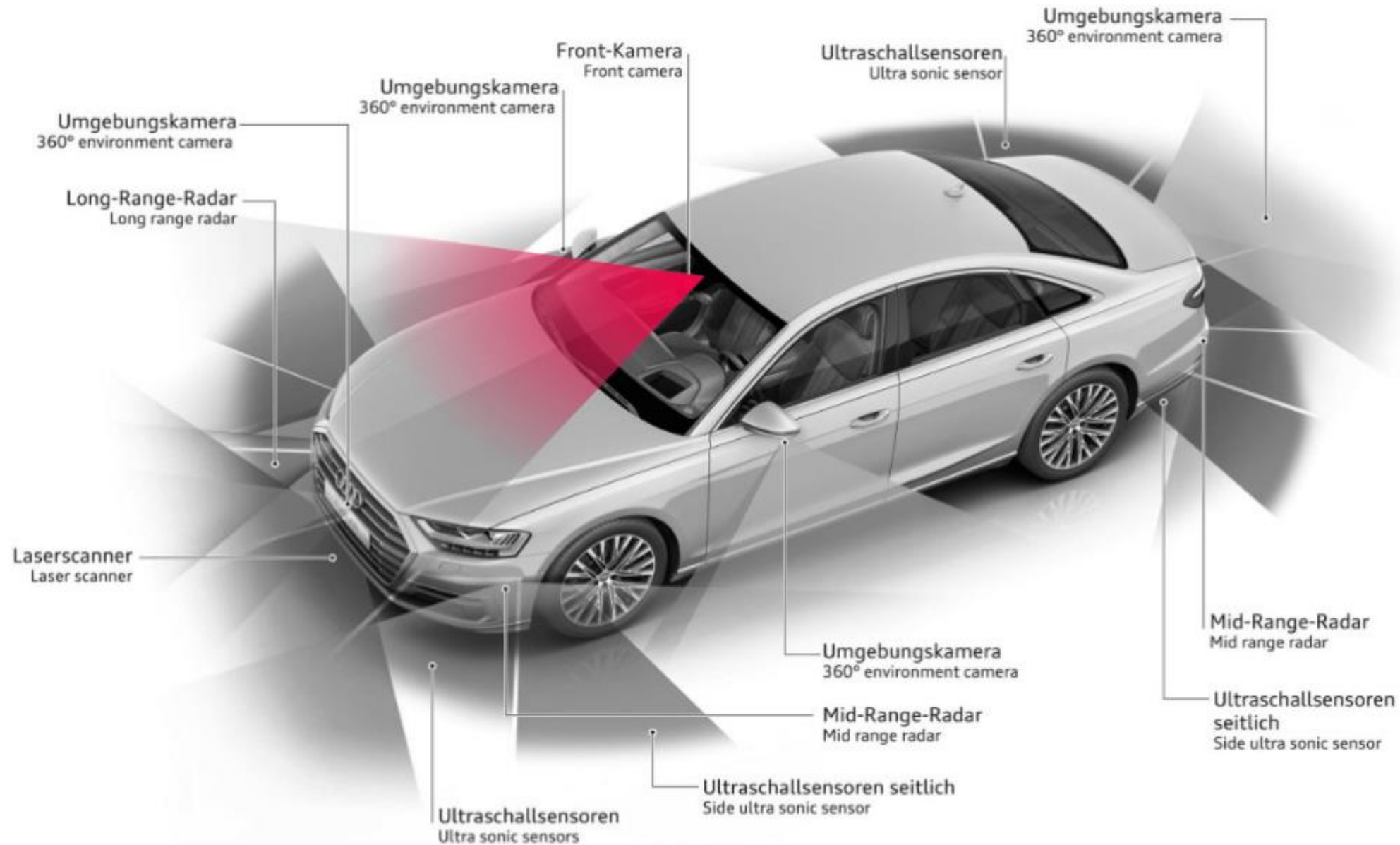


Fig. 1: Sensor alignments, Audi [\[Link\]](#)

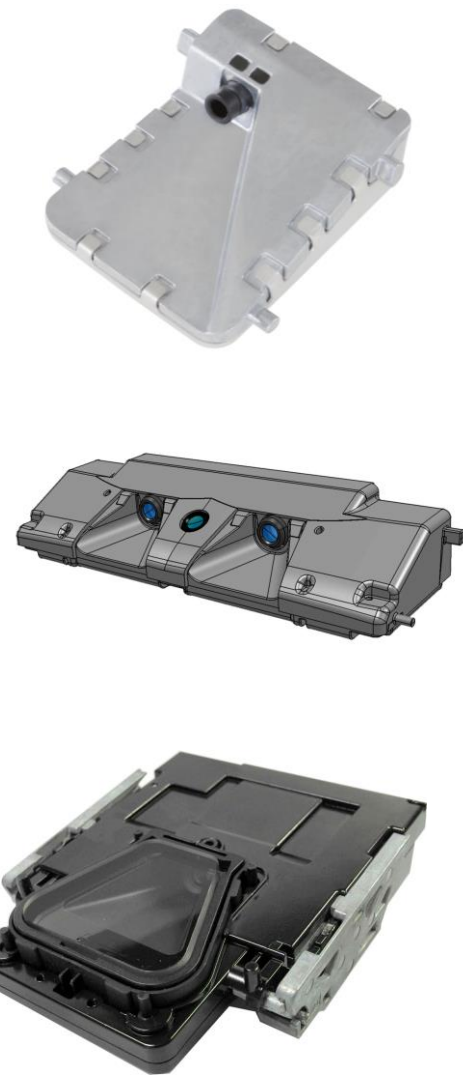


Fig. 2: Various front cameras [\[Link\]](#)



Vid. 1: Tesla AI based environmental perception [\[Link\]](#)

Tesla Stops Putting Radar Sensors in New Model S and Model X EVs

These Teslas join the Model 3 and Model Y in relying solely on cameras. Temporarily, the Autosteer feature may be limited to 80 mph and adaptive cruise control may require a longer minimum following distance.

BY SEBASTIAN BLANCO FEB 28, 2022



Tesla AI chief explains why self-driving cars don't need lidar

By Ben Dickson - June 28, 2021



9 min read



Fig. 3: Various blog posts regarding Tesla's strategic decision wrt. perception.

1. Optical basics

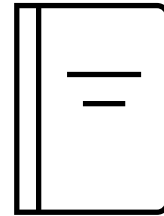
1. Human color perception
2. Color images and perception
3. Pinhole camera model and its use case

2. Color spaces and representations

1. Standard color space XYZ
2. RGB
3. HSI

3. Digital image processing

1. Basics of features (edges, lines, masks, ...)
2. Color segmentation
 1. Road segmentation
 2. Shadow optimization
3. Segmentation by color models
4. Monocular distance estimation



Lecture information

1. Duration approx. 90 minutes
2. Code can be found on [GitHub](#)



Real world use case for this lecture

1. Free space segmentation
2. Object detection and object localization

1. Optical basics

1. Human color perception
2. Color images and perception
3. Pinhole camera model and its use case

2. Color spaces and representations

1. Standard color space XYZ
2. RGB
3. HSI

3. Digital image processing

1. Basics of features (edges, lines, masks, ...)
2. Color segmentation
 1. Road segmentation
 2. Shadow optimization
3. Segmentation by color models
4. Monocular distance estimation

- The human eye serves as a reference for the pinhole camera model
 - *Cornea* and *lens* collect the incident light
 - Light subsequently passes through the transparent *vitreous humour* onto the *retina*
- Colors are registered by the human eye as **distribution of intensity over the wavelength** (λ)
- **Color is not a feature** or condition of an object or material. The same object can evoke in different color sensations depending on the environmental conditions
- Color perception is essentially created by the interaction
 1. of the spectrum and the direction of the illumination,
 2. the spectral reflection, scattering, absorption and transmission properties of the observed object
 3. and the biological conditions of the observer
- There are five different light-sensitive sensor-cells in the *retina*
 1. Rod cell (very sensitive to light, enable vision at very low light intensity)
 2. **Cone cells** (three types of cones(S, M, L), which differ in their spectral sensitivity)
 3. Photoreceptor cell

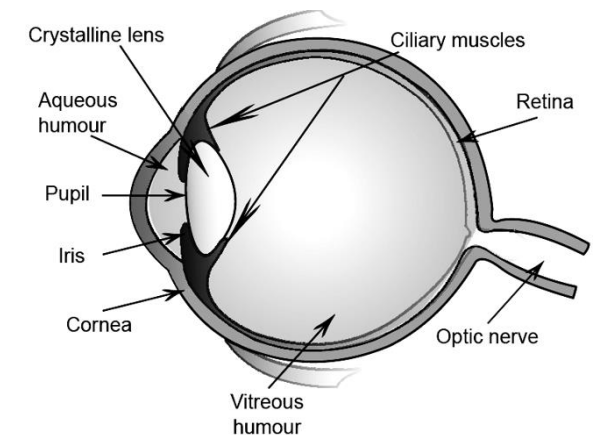


Fig. 4: Human eye [\[Link\]](#)

1. Optical basics | Color images and perception

- Optoelectronic **matrix sensors** of the type "Charge-Couple Device" (**CCD**) or "Complementary Metal Oxide Semiconductor" (**CMOS**) convert the electromagnetic radiation emitted by each different wavelength ranges from each individual object in a scene into an analog signal
- By means of a so-called **frame grabber**, which transmits the signals to an analog-to-digital converter after reading and amplifying. The resulting digital signal is fed to the **image processor**
- Cameras obtain their ability to perceive colors by using wavelength-specific sensitivities of the individual photodiodes. In addition to the application of **color filters**, **micro lenses** placed on each photodiode represent a method for obtaining color information
- Bayer patterns (*RGGB*), but also *RIIB* and *RCCC* patterns are available [Win15]

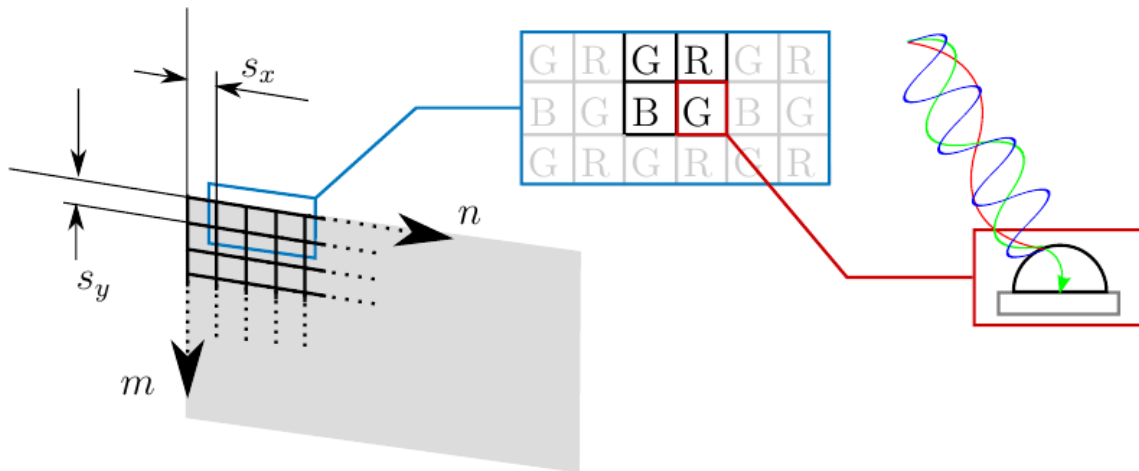
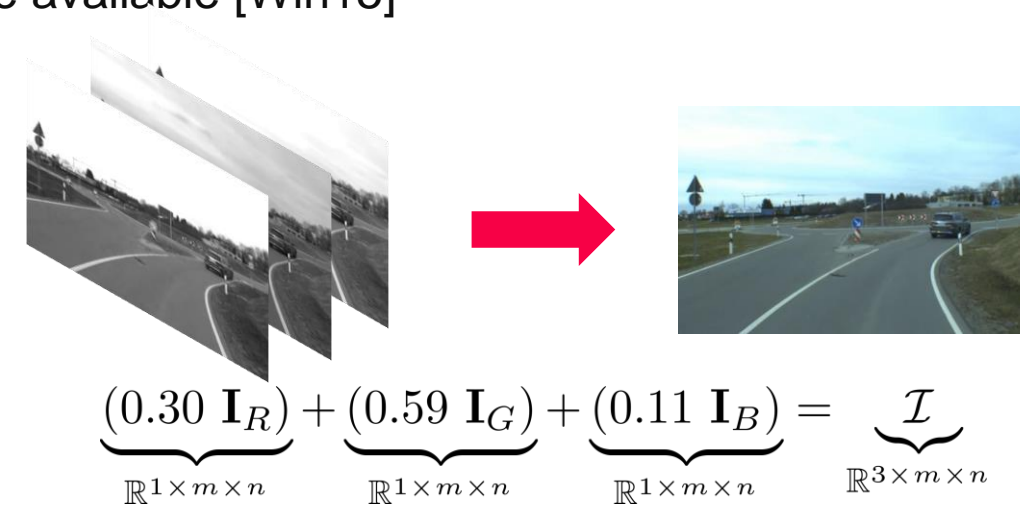


Fig. 5: Image plane with Bayer pattern



$$\underbrace{(0.30 \mathbf{I}_R)}_{\mathbb{R}^{1 \times m \times n}} + \underbrace{(0.59 \mathbf{I}_G)}_{\mathbb{R}^{1 \times m \times n}} + \underbrace{(0.11 \mathbf{I}_B)}_{\mathbb{R}^{1 \times m \times n}} = \underbrace{\mathbf{I}}_{\mathbb{R}^{3 \times m \times n}}$$

Fig. 5a: Image composition from R,G, and B

CCD

- Capacitor is charged by the photodiode
- The charges are read out by means of the so-called frame transfer
- For this purpose, the total **charges** of each capacitor are either shifted into a **darkened area** or, in the case of the so-called "full frame transfer" (where the entire sensor must be darkened), a two-stage readout routine is started directly from there
- In the first stage, the **charges are shifted down** one line by a **vertical shift register**, while in the second stage the currents, which are now in the horizontal shift register, are read out serially
- The process is repeated n times for n lines
- CCD is sensitive for so-called **blooming** artifacts
- If a pixel is exposed too much, this charge quantity is exceeded, and the cell releases the excess charges to the neighboring cells

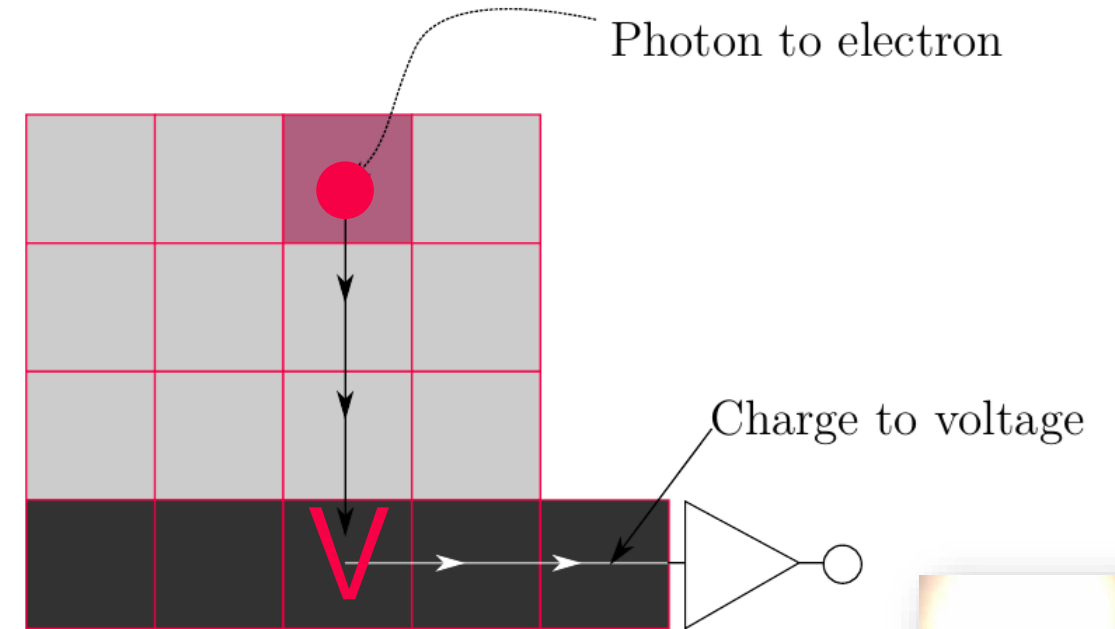


Fig. 6: CCD flow

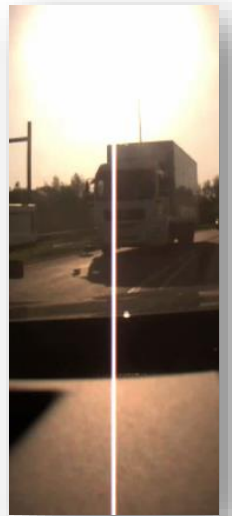


Fig. 6a: Blooming artefact

CMOS

- With the CMOS sensor the **image information can be read out simultaneously** with several channels in parallel and in any order
- This allows several **different regions of interest** on the sensor plane
- By using integrated circuits, it is possible to provide **each pixel with its own amplifier and evaluation electronics**
- A sensor equipped with integrated circuits is called an active pixel sensor (APS)
- Furthermore, the sensors differ in that a **CMOS sensor has a higher dynamic range** of up to 120 dB, while the CCD sensor has only about 50 dB gray scale resolution
- Blooming artifacts do not (that often) occur here, but the so-called **rolling shutter effect** can occur
- To weaken this effect, a **global shutter** was developed for all pixels simultaneously

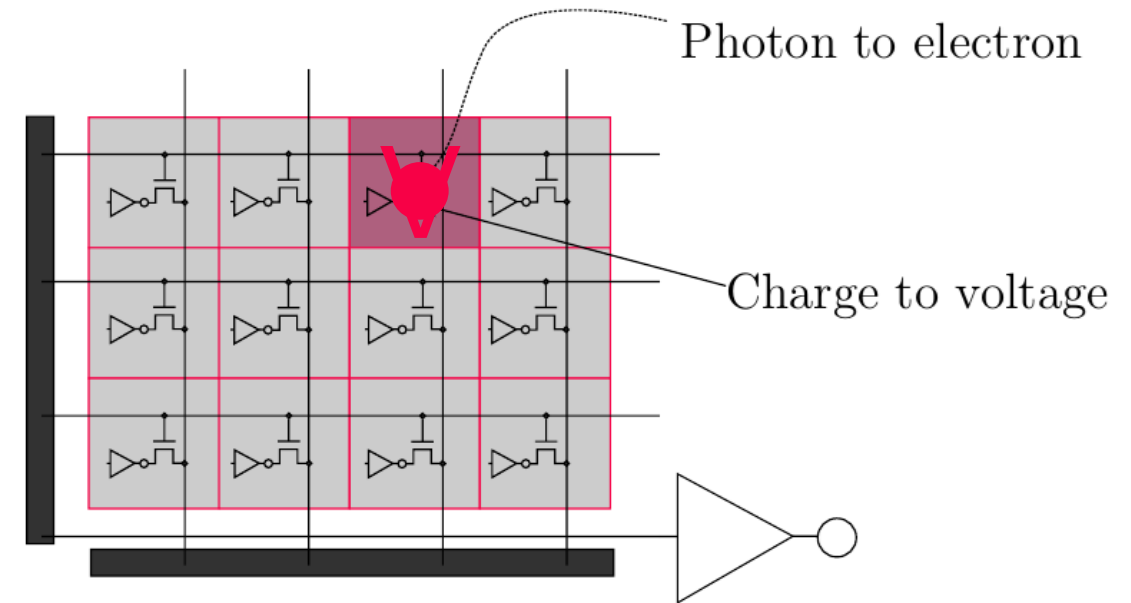
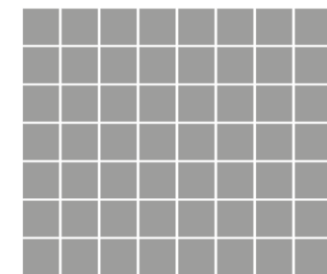


Fig. 7: CMOS flow



Rolling Shutter



Global Shutter

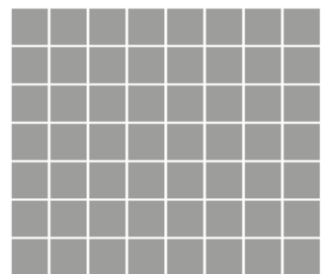
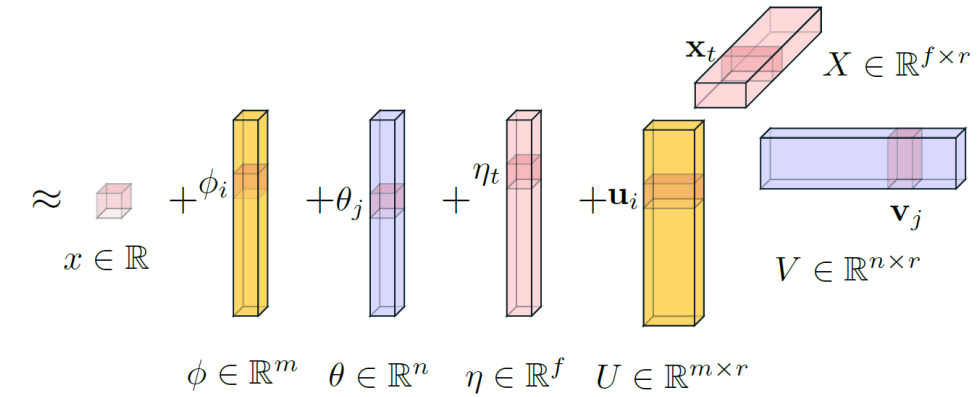
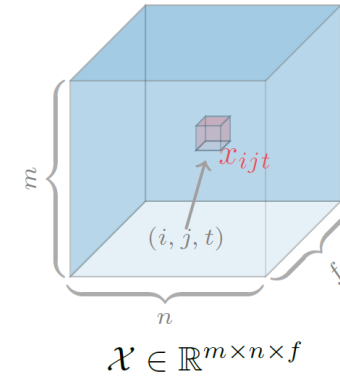


Fig. 7: Rolling Shutter effect and Global Shutter vs. Rolling Shutter

1. Optical basics | Nomenclature

- Scalars: $x \in \mathbb{R}^1$ $x = 1$
- Vectors: $\mathbf{x} \in \mathbb{R}^{1 \times n}$ $\mathbf{x} = [1, 2, 3, 4]$
- Matrices: $\mathbf{X} \in \mathbb{R}^{m \times n}$ $\mathbf{X} = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \end{bmatrix}$



- Orthonormal vector: $\mathbf{R} = \text{dot}(u, v) = \underbrace{u^T v = 0}_{\text{orthogonal}} \wedge \underbrace{u^T u = v^T v = 1}_{\text{unit length}}$
- Characteristics: $\mathbf{R}^{-1} = \mathbf{R}^T$
 $\mathbf{R}^T \mathbf{R} = \mathbf{R} \mathbf{R}^T = \text{Id}$

1. Optical basics | Linear camera model

- The most often used model for bringing three-dimensional spatial coordinates (X, Y, Z) into two-dimensional pixel coordinates $(u, v$ or $x_c, y_c)$ is represented by the **pinhole camera model** [\[Link\]](#)
- To estimate the model, a so-called **camera calibration** is required
- Camera calibration consist of two major parts:
 1. **Position and orientation of the camera** wrt. to world coordinate frame [external, or **extrinsic** calibration]
 2. **Internal parameters** to map points in the world to the image plane such as the focal length [**intrinsic** calibration]
- Camera calibration is about to determine the intrinsic and extrinsic parameters of the camera/lens-combination

1. Optical basics | Linear camera model

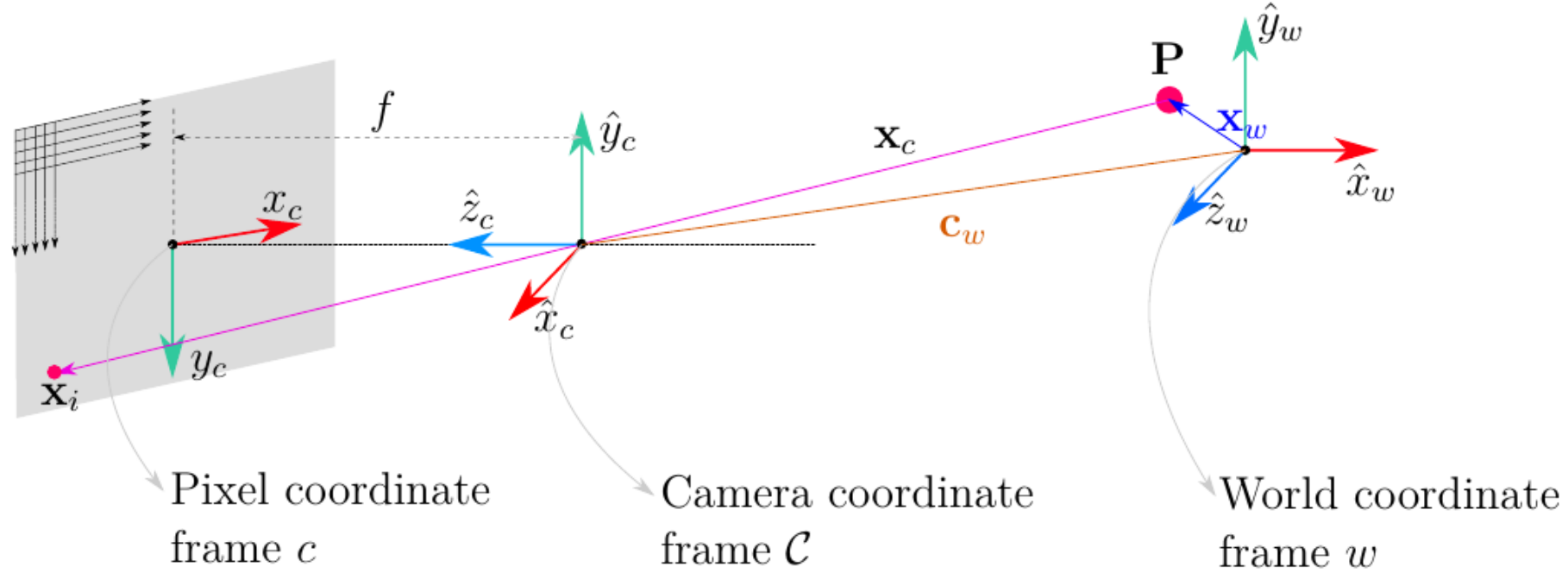
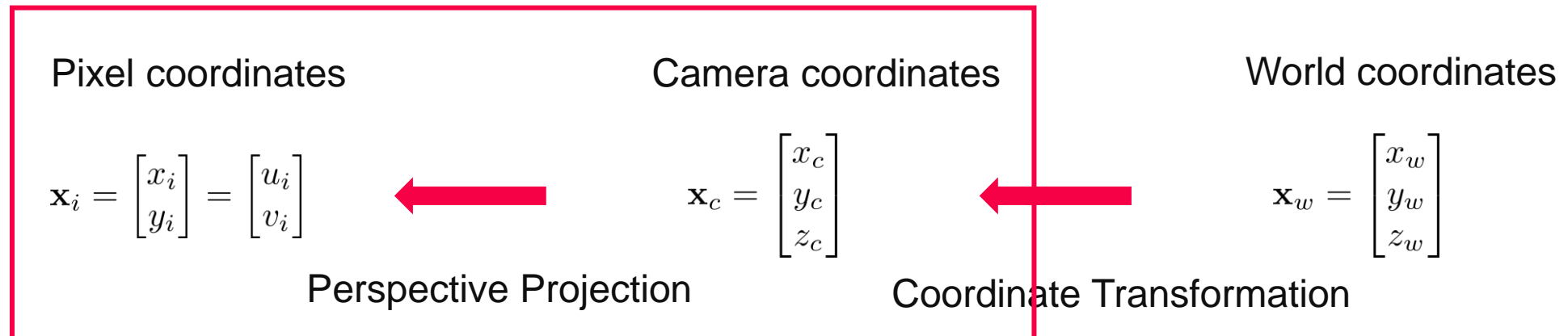
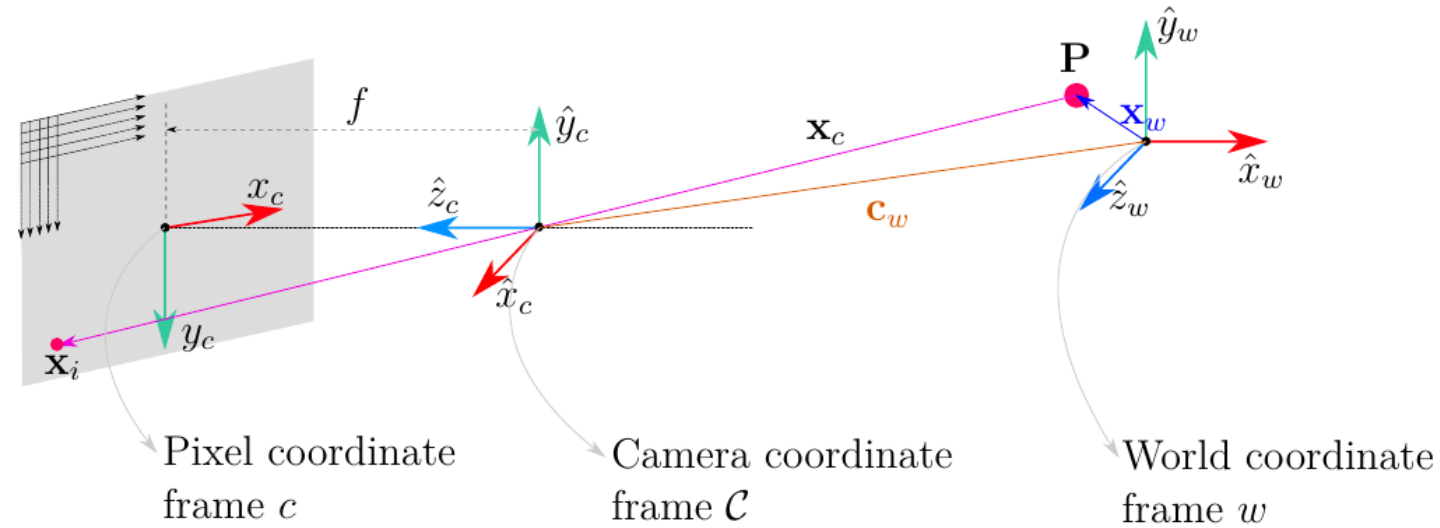


Fig. 8: Pinhole model and world coordinates



1. Optical basics | Perspective Projection

14



We know from trigonometry:

$$\frac{x_i}{f} = \frac{x_c}{z_c} \quad \frac{y_i}{f} = \frac{y_c}{z_c}$$

And therefore:

$$x_i = f \frac{x_c}{z_c} \quad y_i = f \frac{y_c}{z_c}$$

1. Optical basics | Image Plane to Pixel Mapping

- Using the **pixel size** s (usually in μm), a **mapping between the metric plane and the pixel plane** is possible
- Modern sensors have pixel sizes of around $3\text{-}5\ \mu\text{m}$
- The pixel density is called m_x resp. m_y and given in pixel/mm $\rightarrow 1/0.005 \rightarrow 200\ \text{pixel/mm}$

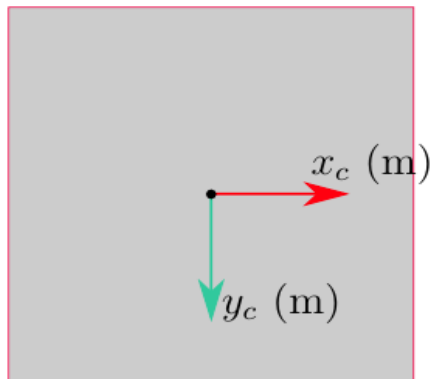


Fig. 9a: Image plane

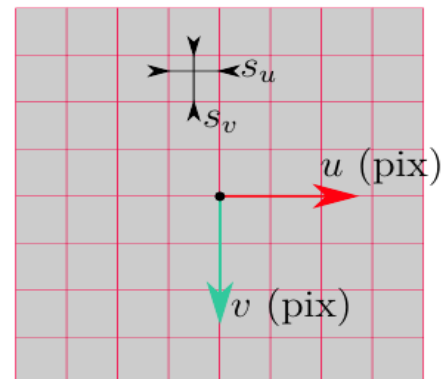


Fig. 9b: Image Sensor/Pixel plane

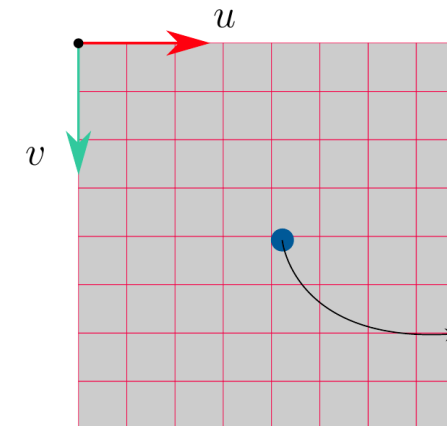


Fig. 9c: Principal point

o_u, o_v OR o_x, o_y

Principal point:
Where the optical axis intersects the image plane

- Hence:

$$u = m_x x_i = m_x f \frac{x_c}{z_c}$$

$$v = m_y y_i = m_y f \frac{y_c}{z_c}$$

$$u = m_x x_i = m_x f \frac{x_c}{z_c} + o_x$$

$$v = m_y y_i = m_y f \frac{y_c}{z_c} + o_y$$

1. Optical basics | Perspective Projection

- Given the Perspective Projection equation

$$u = m_x f \frac{x_c}{z_c} + o_x \quad v = m_y f \frac{y_c}{z_c} + o_y$$

- We can bring down $m_x f$ to f_x resp. f_y

$$u = f_x \frac{x_c}{z_c} + o_x \quad v = f_y \frac{y_c}{z_c} + o_y$$

- The parameters f_x and f_y are so-called **effective focal length** (in pixels) in each direction

$$u = f_x \frac{x_c}{z_c} + o_x \quad v = f_y \frac{y_c}{z_c} + o_y$$

- The unknown (**red**) parameters in the Perspective Projection are the so-called intrinsic parameters (represents the cameras internal geometry) and must be found by calibration
- The model is Non-Linear (**green**) but we can use Homogenous Coordinates to linearize the model

1. Optical basics | Homogenous Coordinates

- In Homogenous Coordinates, we use **fictitious parameters** (\tilde{w}) to scale and normalize a given point
- For that purpose, we extend a given point ($\mathbf{x} \in \mathbb{R}^n$) by the scaling parameter into $\tilde{\mathbf{x}} \in \mathbb{R}^{n+1}$

$$\mathbf{u} = (u, v) \xrightarrow[\text{Coordinates}]{\text{Homogenous}} \tilde{\mathbf{u}} = (\tilde{u}, \tilde{v}, \tilde{w})$$

- Whereby with $u = \frac{\tilde{u}}{\tilde{w}}$ and $v = \frac{\tilde{v}}{\tilde{w}}$ we get:

$$\mathbf{u} \equiv \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \equiv \begin{bmatrix} \tilde{w}u \\ \tilde{w}v \\ \tilde{w} \end{bmatrix} \equiv \begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} = \tilde{\mathbf{u}}$$

- This holds also for n -dimensional data!

$$\mathbf{x} \equiv \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \equiv \begin{bmatrix} \tilde{w}x \\ \tilde{w}y \\ \tilde{w}z \\ \tilde{w} \end{bmatrix} \equiv \begin{bmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{z} \\ \tilde{w} \end{bmatrix} = \tilde{\mathbf{x}}$$

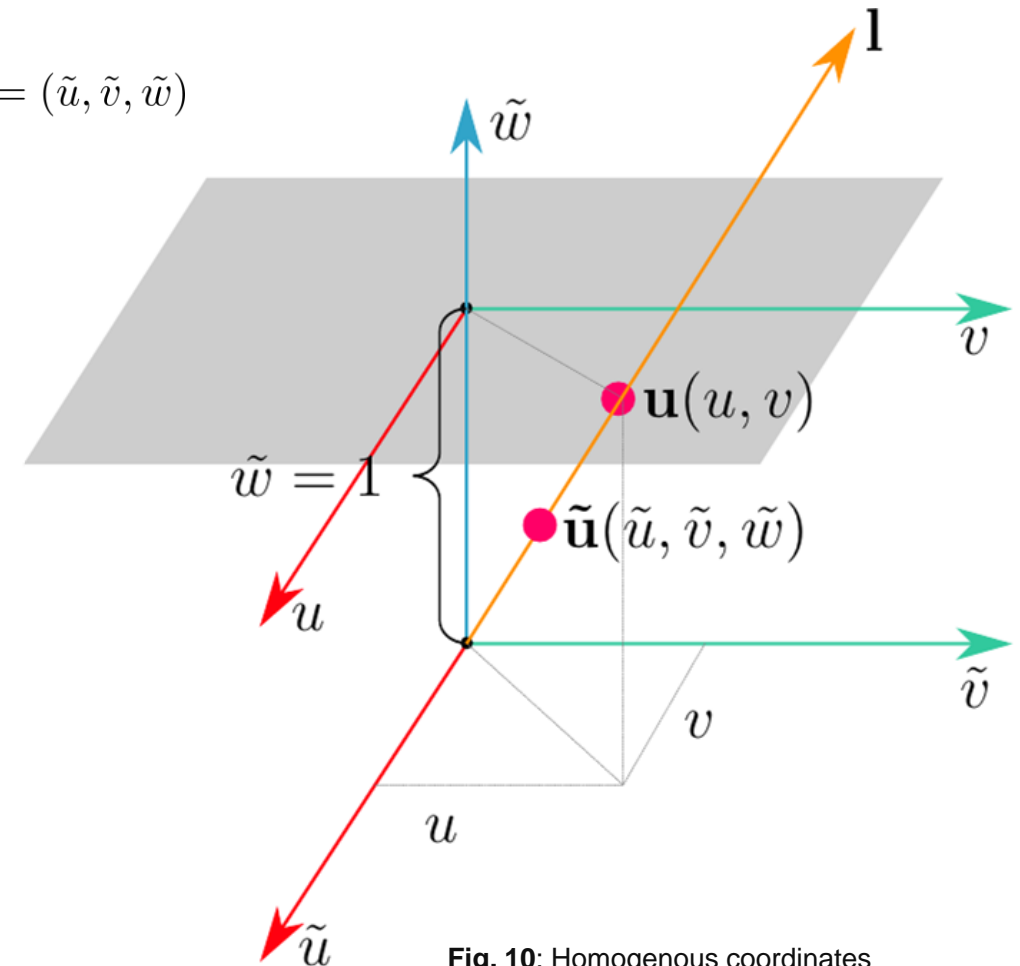


Fig. 10: Homogenous coordinates

1. Optical basics | Perspective Projection

- Given the Perspective Projection equation:

$$u = f_x \frac{x_c}{z_c} + o_x \quad v = f_y \frac{y_c}{z_c} + o_y$$

- We can use Homogenous coordinates:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \equiv \begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} \equiv \begin{bmatrix} z_c u \\ z_c v \\ z_c \end{bmatrix} = \begin{bmatrix} f_x x_c + z_c o_x \\ f_y y_c + z_c o_y \\ z_c \end{bmatrix} = \underbrace{\begin{bmatrix} f_x & 0 & o_x & 0 \\ 0 & f_y & o_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\text{Intrinsic Matrix } \mathbf{M}_{\text{int}}} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}$$

- We get this from:

$$u = f_x \frac{x_c}{z_c} + o_x \quad | \cdot z_c$$

$$z_c u = f_x x_c + z_c o_x$$

- Given the Intrinsic Matrix:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \equiv \begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} = \begin{bmatrix} f_x & 0 & o_x & 0 \\ 0 & f_y & o_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}$$

- We can decompose the Intrinsic Matrix to the so-called **Camera matrix (K)** :

$$\mathbf{K} = \begin{bmatrix} f_x & 0 & o_x \\ 0 & f_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \xrightarrow[\text{Triangular}]{\text{Upper Right}} \mathbf{K} = \begin{bmatrix} f_x & 0 & o_x \\ & f_y & o_y \\ & & 1 \end{bmatrix}$$

- So that, we see:

$$\mathbf{M}_{int} = [\mathbf{K} | \mathbf{0}] = \begin{bmatrix} f_x & 0 & o_x & 0 \\ 0 & f_y & o_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

$$\tilde{\mathbf{u}} = [\mathbf{K} | \mathbf{0}] \tilde{\mathbf{x}}_c = \mathbf{M}_{int} \tilde{\mathbf{x}}_c$$

Summery

- Using optical geometry enables us to find **correlation between a 3D point in camera coordinates and 2D pixel coordinates**

$$x_i = f \frac{x_c}{z_c} \quad y_i = f \frac{y_c}{z_c}$$

- Using the sensor information (pixel size) enables us to **map the projected point to pixel coordinates**

$$u = m_x f \frac{x_c}{z_c} + o_x \quad v = m_y f \frac{y_c}{z_c} + o_y$$

- By means of the geometry, we can construct a Non-Linear camera model

$$u = f_x \frac{x_c}{z_c} + o_x \quad v = f_y \frac{y_c}{z_c} + o_y$$

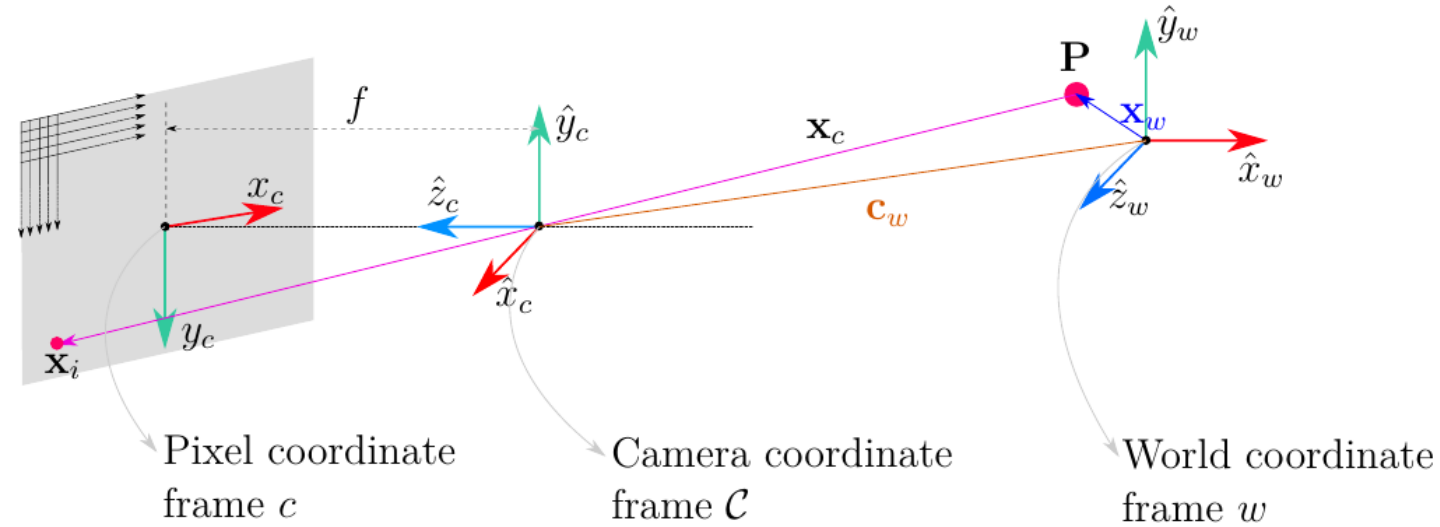
- by using Homogenous coordinates, we end up with a **linear camera model**

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \equiv \begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} = \begin{bmatrix} f_x & 0 & o_x & 0 \\ 0 & f_y & o_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}$$

- We use the **Intrinsic Matrix** to represent a point in camera coordinates (3D) in pixels (2D)

$$\tilde{\mathbf{u}} = [\mathbf{K} | \mathbf{0}] \tilde{\mathbf{x}}_c = \mathbf{M}_{\text{int}} \tilde{\mathbf{x}}_c$$

1. Optical basics | Linear camera model



Pixel coordinates

$$\mathbf{x}_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix} = \begin{bmatrix} u_i \\ v_i \end{bmatrix}$$

Camera coordinates

$$\mathbf{x}_c = \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix}$$

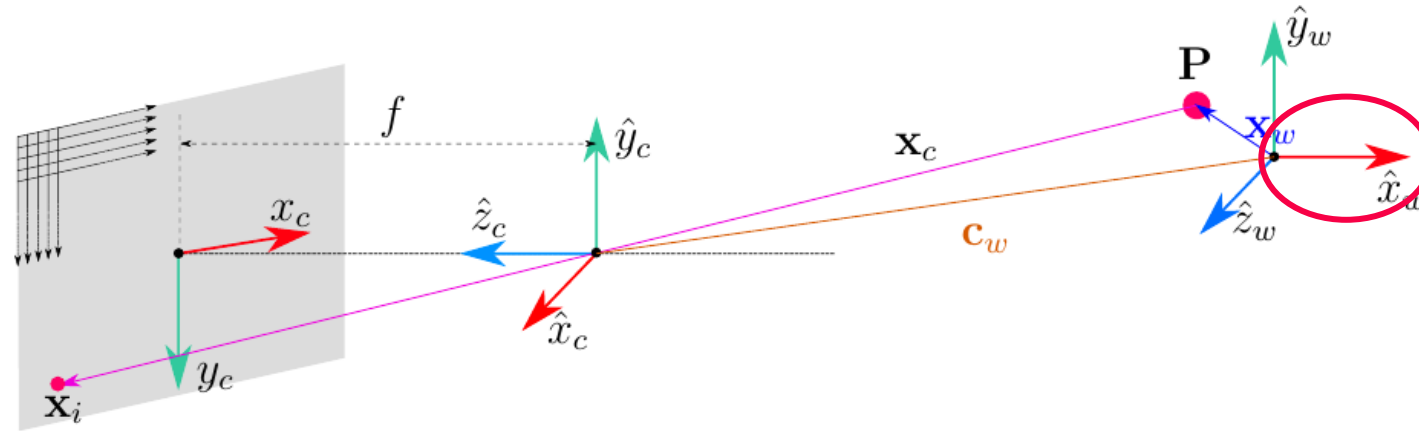
World coordinates

$$\mathbf{x}_w = \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix}$$

Perspective Projection

Coordinate Transformation

1. Optical basics | Extrinsic Camera Calibration



Position \mathbf{c}_w and the Rotation \mathbf{R} of the camera in the world frame are the so-called extrinsic parameters

$$\mathbf{R} = \underbrace{\begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}}_{\text{Orthonormal}} \quad \left| \quad \begin{array}{l} \rightarrow \text{Row 1 : Correlation between } \hat{x}_w \text{ and } \hat{x}_c \\ \rightarrow \text{Row 2 : Correlation between } \hat{y}_w \text{ and } \hat{y}_c \\ \rightarrow \text{Row 3 : Correlation between } \hat{z}_w \text{ and } \hat{z}_c \end{array} \right.$$

$$\mathbf{t} = -\mathbf{R}\mathbf{c}_w$$

$$\mathbf{x}_c = \mathbf{R}(\mathbf{x}_w - \mathbf{c}_w) = \mathbf{R}\mathbf{x}_w - \mathbf{R}\mathbf{c}_w = \mathbf{R}\mathbf{x}_w + \mathbf{t}$$

$$\mathbf{x}_c = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$$

1. Optical basics | Extrinsic Camera Calibration

- Using homogenous coordinates:

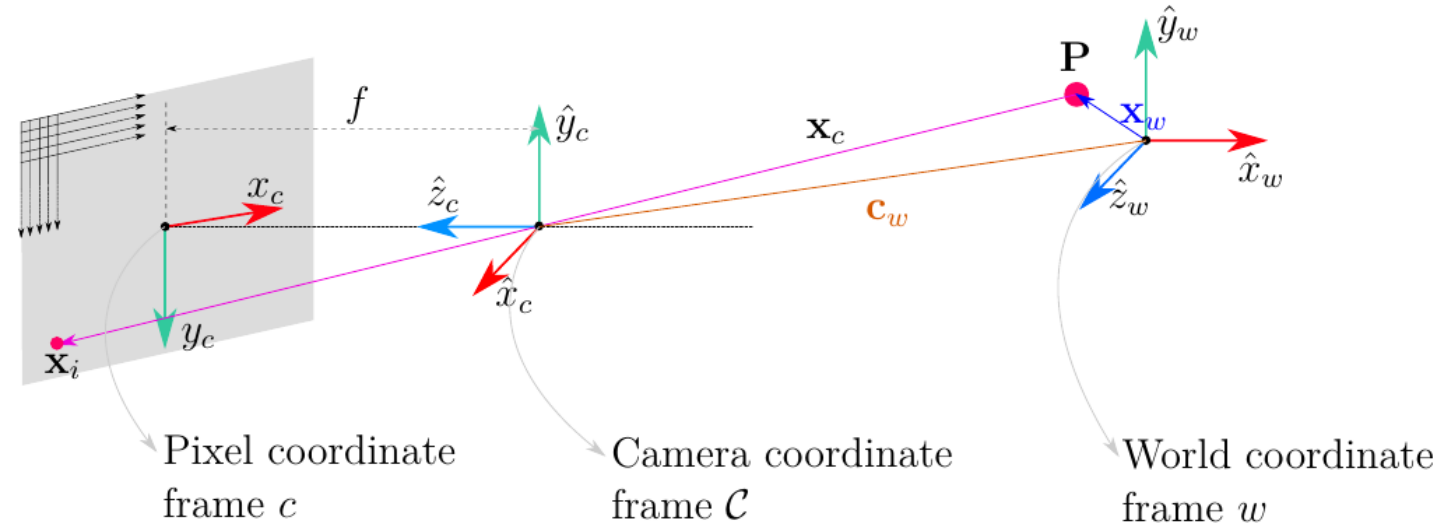
$$\mathbf{x}_c = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$$

$$\tilde{\mathbf{x}}_c = \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{\text{Extrinsic Matrix } \mathbf{M}_{\text{ext}}} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}$$

$$\mathbf{M}_{\text{ext}} = \begin{bmatrix} \mathbf{R}^{3 \times 3} & \mathbf{t} \\ \mathbf{0}^{1 \times 3} & 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\tilde{\mathbf{x}}_c = \mathbf{M}_{\text{ext}} \tilde{\mathbf{x}}_w$$

1. Optical basics | Linear camera model



Pixel coordinates

$$\mathbf{x}_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix} = \begin{bmatrix} u_i \\ v_i \end{bmatrix}$$

Camera coordinates

$$\mathbf{x}_c = \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix}$$

World coordinates

$$\mathbf{x}_w = \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix}$$

Perspective Projection

Coordinate Transformation

Camera to Pixel:

$$\begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} = \begin{bmatrix} f_x & 0 & o_x & 0 \\ 0 & f_y & o_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}$$

$$\tilde{\mathbf{u}} = \mathbf{M}_{\text{int}} \tilde{\mathbf{x}}_c$$

World to Camera:

$$\tilde{\mathbf{x}}_c = \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}$$

$$\tilde{\mathbf{x}}_c = \mathbf{M}_{\text{ext}} \tilde{\mathbf{x}}_w$$

$$\tilde{\mathbf{u}} = \mathbf{M}_{\text{int}} \mathbf{M}_{\text{ext}} \tilde{\mathbf{x}}_w = \mathbf{P} \tilde{\mathbf{x}}_w$$

$$\mathbf{P} = \begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}$$

- To calibrate the camera, we must find the 12 missing parameters of the Projection Matrix
- For that approach, so-called PnP-Algorithm is used, but not considered in this lecture

1. Optical basics

1. Human color perception
2. Color images and perception
3. Pinhole camera model and its use case

2. Color spaces and representations

1. Standard color space XYZ
2. RGB
3. HSI

3. Digital image processing

1. Basics of features (edges, lines, masks, ...)
2. Color segmentation
 1. Road segmentation
 2. Shadow optimization
3. Segmentation by color models
4. Monocular distance estimation

2. Color spaces and representations | Standard color space (XYZ)

- As early in 1931, the *International Commission on Illumination* (C.I.E., french. *Commission Internationale de l'Éclairage*) standardized the procedure for measuring color valences and brought them into a standard color space
- With the introduction of that, a standardization of the “white” was also introduced. So that a **white color** stimulus **has three identical color values** corresponding to $R_{CIE} = G_{CIE} = B_{CIE}$
- The color values $\bar{r}, \bar{g}, \bar{b}$ are the weightings of the C.I.E primary light sources

$$R_{CIE} = \int_{380}^{780} I(\lambda) \bar{r}(\lambda) d\lambda \quad G_{CIE} = \int_{380}^{780} I(\lambda) \bar{g}(\lambda) d\lambda \quad B_{CIE} = \int_{380}^{780} I(\lambda) \bar{b}(\lambda) d\lambda$$

- By means of a transformation matrix, we obtain the 3D color space XYZ

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = A \begin{pmatrix} R_{CIE} \\ G_{CIE} \\ B_{CIE} \end{pmatrix} \quad A \in \mathbb{R}^{3 \times 3} \quad \text{with} \quad A = \begin{pmatrix} 0.49000 & 0.31000 & 0.20000 \\ 0.17697 & 0.81240 & 0.01063 \\ 0.00000 & 0.01000 & 0.99000 \end{pmatrix}$$

- Based on a defined standard color space, various **perceptually** as well as **technically** oriented color spaces can be derived and transformed into each other

2. Color spaces and representations | XYZ & sRGB color space

- The **chrominance** (Farbigkeit) can be transformed independently of the brightness into a two-dimensional space

$$x = \frac{X}{X+Y+Z} \quad y = \frac{Y}{X+Y+Z}$$

- Thus, a color is uniquely described by its chrominance values in x and y and its luminance value Y
- The CIE standard chromaticity diagram is the representation of the color in the two-dimensional x-y-plane

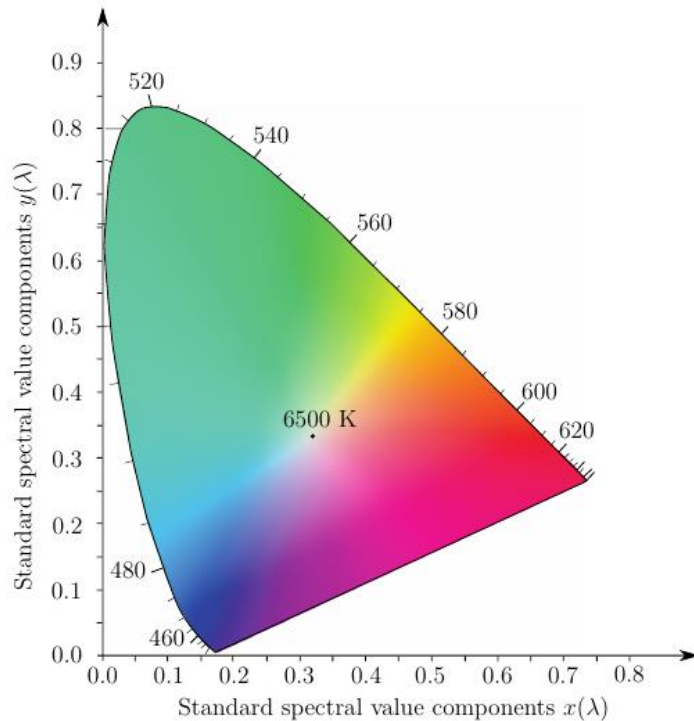


Fig. 11: CIE-standard in x-y-plane

$$\begin{pmatrix} R_S \\ G_S \\ B_S \end{pmatrix} = A_{sRGB} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \quad A_{sRGB} \in \mathbb{R}^{3 \times 3}$$



$$A_{sRGB} = \begin{pmatrix} 3.2410 & -1.5374 & -0.4986 \\ -0.9692 & 1.8760 & 0.0416 \\ 0.0556 & -0.2040 & 1.0570 \end{pmatrix}$$

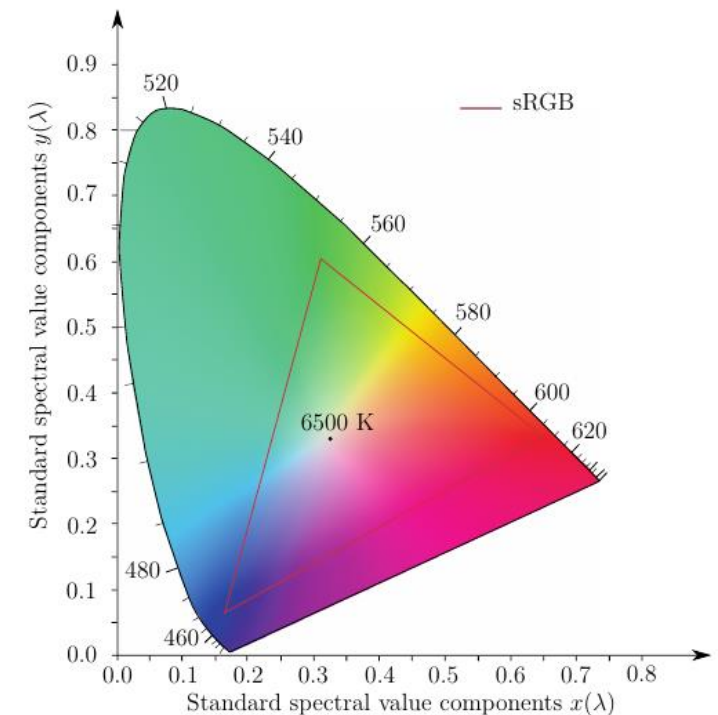


Fig. 12: CIE-standard in x-y-plane with sRGB gamut

2. Color spaces and representations | HSI color space

- The components in the color space HSI correspond to **h**ue, **s**aturation and **i**ntensity and are represented as a double cylinder
- The colors in this color space are represented in **accordance with human vision** and thus this color space is assigned to the **perception-oriented color spaces**
- This color space is **illumination invariant** in the channels **H** and **S** (the same color representation for cloudy, sunny, rainy, backlight, ... situations)

$$I_i = \frac{R_i + G_i + B_i}{3} \quad S_i = 1 - \frac{\min(R_i, G_i, B_i)}{I_i}$$

$$H = \begin{cases} \theta, & B_i \leq G_i \\ 360 - \theta & \text{else} \end{cases} \quad \text{with} \quad \theta = \cos^{-1} \left(\frac{\frac{1}{2}(R_i - G_i) + (R_i - B_i)}{\sqrt{(R_i - G_i)^2 + (R_i - B_i)(G_i - B_i)}} \right)$$

- Approximation by [CT11] and [SRM04]

$$\begin{pmatrix} I \\ V_1 \\ V_2 \end{pmatrix} = \begin{pmatrix} \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ \frac{-1}{\sqrt{6}} & \frac{-1}{\sqrt{6}} & \frac{2}{\sqrt{6}} \\ \frac{1}{\sqrt{6}} & \frac{-2}{\sqrt{6}} & \frac{1}{\sqrt{6}} \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} \quad H = \tan^{-1} \left(\frac{V_2}{V_1} \right)$$

$$S = \sqrt{V_1^2 + V_2^2}$$

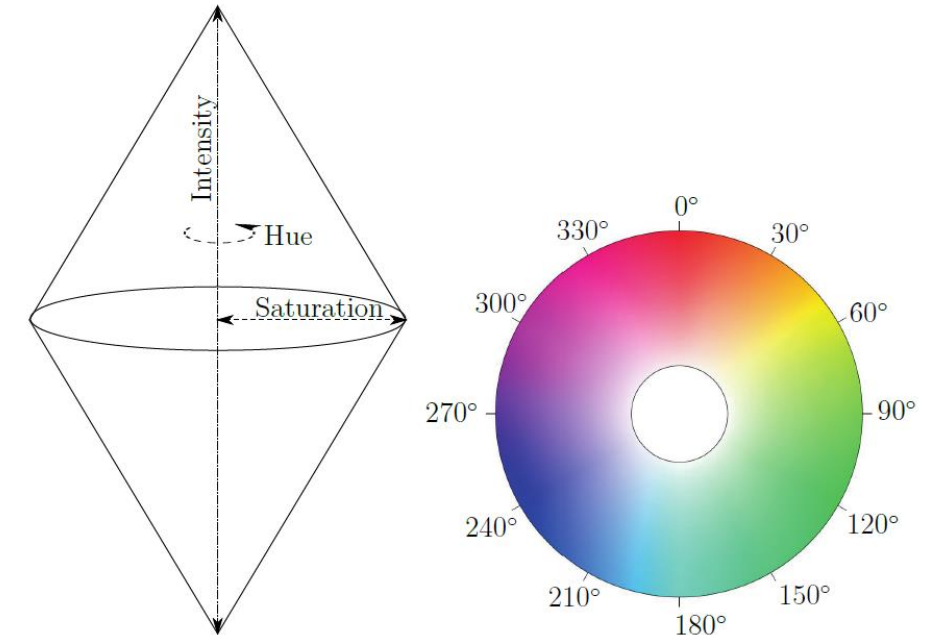


Fig. 14: HSI color space from [Jae12]

1. Optical basics

1. Human color perception
2. Color images and perception
3. Pinhole camera model and its use case

2. Color spaces and representations

1. Standard color space XYZ
2. RGB
3. HSI

3. Digital image processing

1. Basics of features (edges, lines, masks, ...)
2. Color segmentation
 1. Road segmentation
 2. Shadow optimization
3. Segmentation by color models
4. Monocular distance estimation

3. Digital image processing | Color segmentation

- To select similar ranges (in terms of color representation), we can use the illumination invariant color space HSI and its properties
- A selected road pattern with the feature vector $X_p = (H_p, S_p, I_p)^T$ provides a basis on which a pixel element at location i ($X_i = (H_i, S_i, I_i)^T$) is compared to
- By determining the deviation in intensity (Δ_{Int}) as well as chrominance (Δ_C), a threshold segmentation can be realized

$$\Delta_I = |I_p - I_i| \quad \Delta_C = \sqrt{S_p^2 + S_i^2 - 2S_p S_i \cos(\theta)}$$

$$\text{with } \theta = \begin{cases} |H_p - H_i| & \text{if } |H_p - H_i| \leq 180^\circ \\ 360^\circ - |H_p - H_i| & \text{if } |H_p - H_i| > 180^\circ \end{cases}$$

- The final decision rule (kind of softened borders) follows:

$$T_C(t+1) = \bar{\Delta}_C(t) \exp\left(\frac{K}{2}\right) \quad \text{whereby} \quad \bar{\Delta}(\cdot) = \sqrt{\frac{1}{n} \sum_{j=0}^{n-1} \Delta_{(\cdot)}^2(t)}$$

$$T_I(t+1) = \bar{\Delta}_I(t) \exp\left(\frac{K}{2}\right)$$

$$K = -2 \ln\left(\frac{7}{10}\right)$$

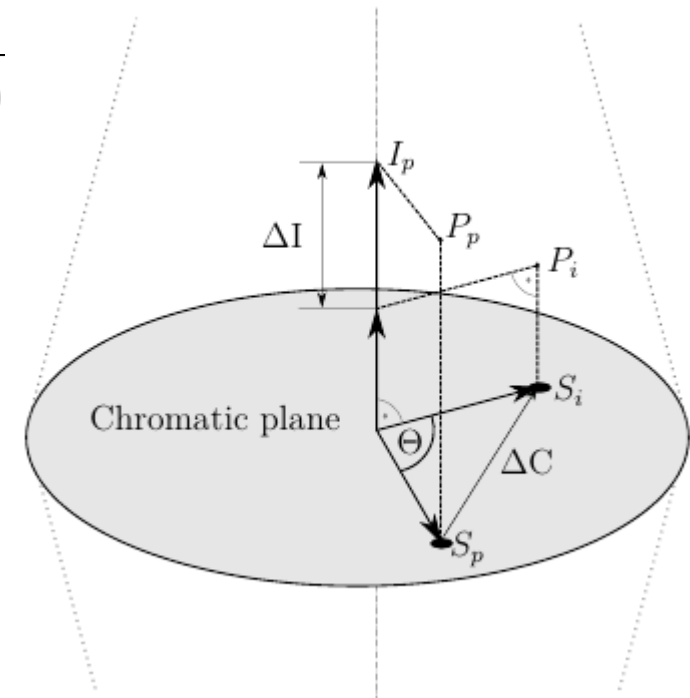


Fig. 15: Color segmentation by means of HIS from [Sot08]

3. Digital image processing | Color segmentation

- A set of n seeds in front of the vehicle provides a road pattern
- Finding similar areas (in terms of color representation) provides a first guess for free road space
- Small optimizations (morphological opening, merging etc.) provides unfiltered free space

Road samples:

```
# ROI road search path
road_example_x = 620
road_example_y = 425
road_example_width = 100
road_example_height = 25
n_samples = 12

update_interval = 15

# Get random points within
road_example_pts = np.array(
    rr_end_y, n_samples),

np.random.randint(rr_start_x

road_example_h = np.mean([
road_example_pts[1, i], 0])
dtype=int)

road_example_s = np.mean([
road_example_pts[1, i], 1] for i in range(0, n_samples)],
dtype=int)

road_example_i = np.mean([img_hsi[road_example_pts[0, i],
road_example_pts[1, i], 2] for i in range(0, n_samples)],
dtype=int)
```



Thresholding:

```
road_example_h*(1 -
road_example_s*(1 -
ad_example_i*(1 -
road_example_h*(1 +
road_example_s*(1 +
ad_example_i*(1 +

img_mask[0:n_h*20_y, :] = 0

# Dilatation
img_mask = cv2.dilate(img_mask, np.ones((3, 3), 'uint8'),
iterations=1)
```

3. Digital image processing | Shadow properties

- Shadows are dark areas where the light from a light source is blocked by an opaque object
- Shaded areas can be detected on images not only on sunny days, but also in diffuse light
- **Shadows are problematic when features are incorrectly extracted** due to the darker areas and thus lead to a false positive feature
- **Shaded areas appear blue** under natural lighting because no direct light from the light source falls on the shaded area, but it is merely illuminated by the prevailing ambient illumination [Chu94, Lyn15].

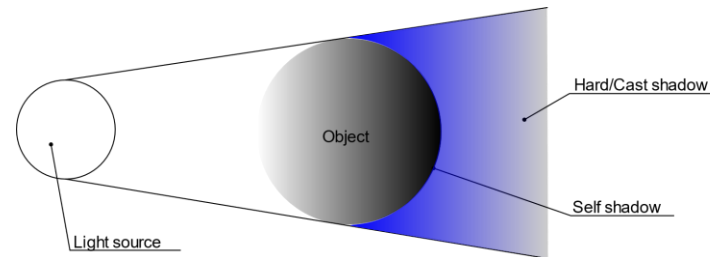


Fig. 16: Shadow and opaque object

- Furthermore, a **shadow region is characterized by a reduced intensity** compared to the environment illumination
- As early as 1808, *Johann Wolfgang von Goethe* assigned the following words to his publication "Zur Farbenlehre" ("On the Theory of Colors"): "At dusk, place a low-burning candle on a white paper; between it and the waning daylight, place a pencil upright, so that the shadow cast by the candle is illuminated by the weak daylight, but cannot be cancelled out, and the shadow will appear of the most beautiful blue."

3. Digital image processing | Shadow properties

- Shadows are dark areas where the light from a light source is blocked by an opaque object
- Shaded areas can be distinguished from the surrounding areas by their lower intensity
- **Shadows are problematic** because they are not a true feature of the scene but a false positive feature
- **Shadows are problematic** because they are not a true feature of the scene but a false positive feature
- **Shaded areas appear dark** because the light source falls on the surrounding area, but it is not on the shaded area [94, Lyn15].
- Furthermore, a **shadow** is a dark area that is not a true feature of the scene but a false positive feature
- As early as 1808, Johann Wolfgang von Goethe published his publication "Zur Farbenlehre" ("On the Theory of Colours") where he states: "On the waning daylight, place a pencil upright, so that the shadow cast by the candle is illuminated by the weak daylight, but cannot be cancelled out, and the shadow will appear of the most beautiful blue."



3. Digital image processing | Color segmentation

- To increase the segmentation quality, it is important that objects that create a shadow due to natural and artificial lighting situations are detected and treated separately
- Using the shadow properties (remember the optical basics), we can identify the shaded areas based on the standard deviation of the previously masked road pixels

$$I_i \leq (I_{\text{road average}} - 2\sigma_{\text{road}})$$

- The color value "Blue" is located at 240° in the HSI circle
- The check for an increased occurrence of blue values in the potential shadow area can be checked to see whether the color value is in the range of $240^\circ \pm 40^\circ$

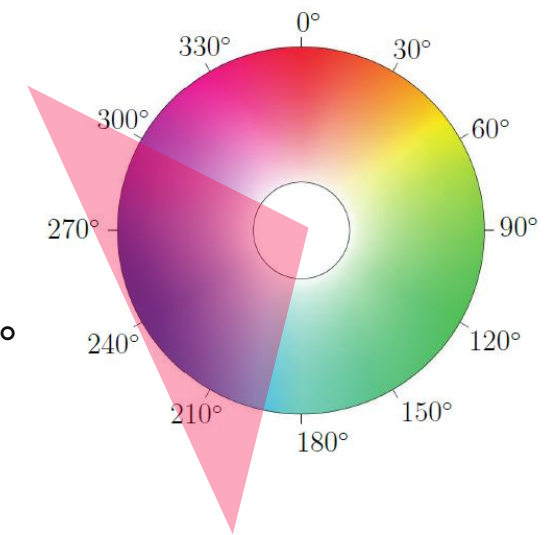


Fig. 17: HIS Chromatic plane with increased blue values

3. Digital image processing | Color segmentation



1. Road pattern is selected in front of the vehicle



2. Road pattern with increased blue values is used to identify shadowed areas



3. Free space is enriched with shadow information

Fig. 18: Shadow identification and segmentation

3. Digital image processing | Color models

- Due to standardization, the traffic signs are (locally) globally in similar colors



Fig. 19: Stop sign, worldwide [\[Link\]](#)

- Using the color representation to create a **generic color model** for further segmentation
- Process:
 1. Extract color patches in various illumination situations and condition
 2. Read the color information for each pixel in each patch
 3. Map the color information in 2D-plane of the color space (for instance in H-S)
 4. Normalize the color model to a floating-point Grayscale image [0;1]
 5. Apply Gaussian kernel for better generalization
 6. Apply the color model for probability-based segmentation

3. Digital image processing | Color models

Color model generation flow

1) Data set

2) Averaging

3) Mapping and Normalizing

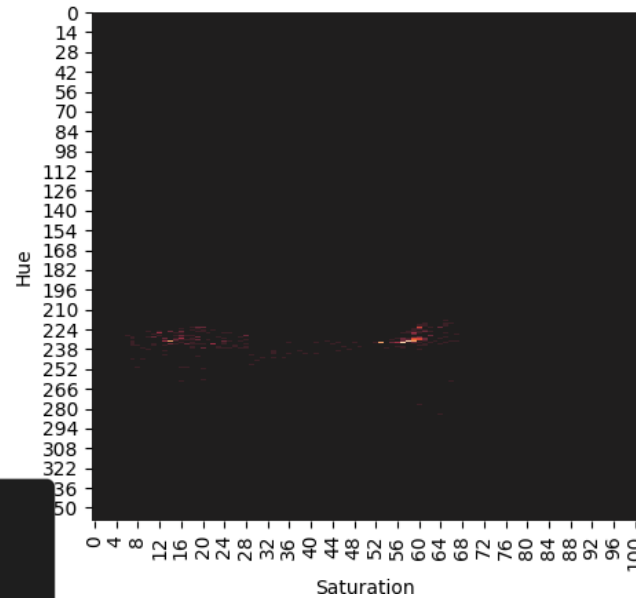
4) Blurring



$X^{i \times m \times n}$

$$H_{idx} = \frac{1}{mn} \sum_{i=0}^{mn} \mathbf{I}(0)$$

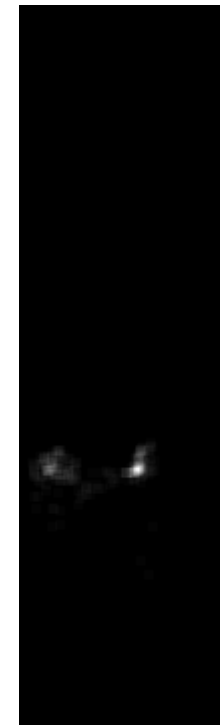
$$S_{idx} = \frac{1}{mn} \sum_{i=0}^{mn} \mathbf{I}(1)$$



$CM \rightarrow [0; 1]$



$CM * K \left(Id_{\frac{1}{25}} \right)^{5 \times 5}$



CM_b

```
for k, file in enumerate(tqdm(files)):
    img = np.asarray(imageio.imread(dataset_path + '/' +
    file))
    img = bgr_2_hsi(img)

    H.append(img[:, :, 0].mean())
    S.append(img[:, :, 1].mean())
    I.append(img[:, :, 2].mean())

    color_model = np.zeros([360, 100])
    for i in range(len(H)):
        color_model[int(H[i])-1, int((S[i]/255)*100)-1] += 1
```

3. Digital image processing | Probability-based segmentation

Applying the model results in a probability image

- Process:

1. Iterate over each pixel (i, j) in the input image I
2. Extract the \mathbf{H} and \mathbf{S} values at the position $I(i, j)$
3. Go into the color model (CM) at the position of \mathbf{H} and \mathbf{S}
4. Extract probability at the position $\text{CM}(\mathbf{H}, \mathbf{S})$
5. Apply on mask at image position $M(i, j)$

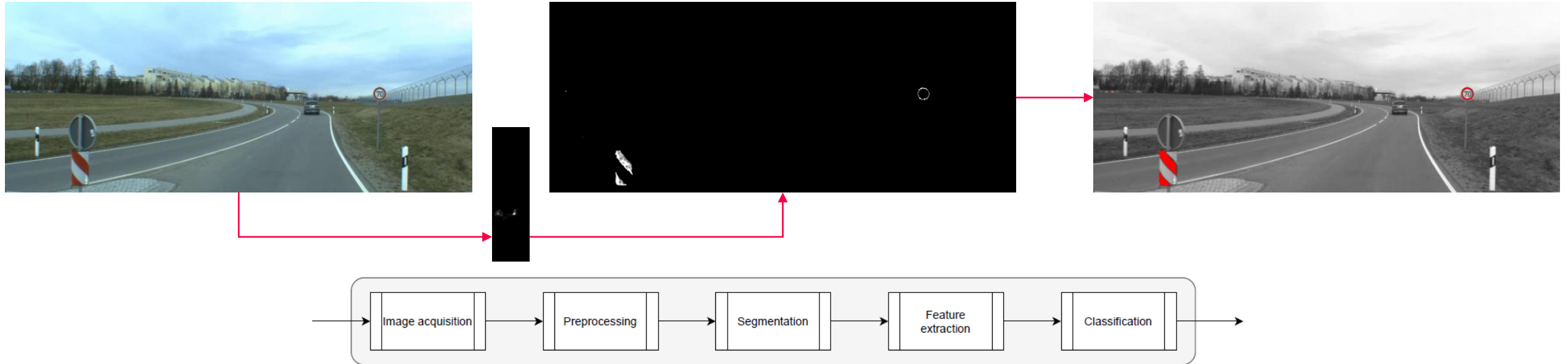


Fig. 20: Image processing pipeline from image acquisition to object classification

3. Digital image processing | Monocular distance estimation

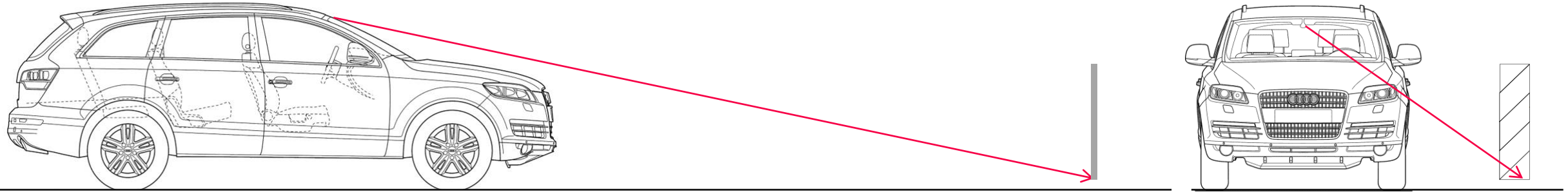


Fig. 21: Visualized visual beam from camera sensor to object, $Z=0$ as restriction.

- Using the linear camera model and pose to estimate the distance of an object
- A-priori knowledge ($Z=0$) required, otherwise we can not solve the equations
- Mathematically, we determine the **intersection of a straight line with a plane**
- We can solve this by inverting the pinhole model

```
def pixel_to_world(self, u,v, Z=0):
    """
    Applies the inverse pinhole model to get world coordinates
    from pixel.
    """

    RMU_inverse = np.linalg.inv(self.calibration.rotation_matrix)
    @ np.linalg.inv(self.calibration.camera_matrix) @ np.array([[u],
    [v], [1]])

    Rt_inverse = np.linalg.inv(self.calibration.rotation_matrix)
    @ self.calibration.translation_vector

    wc = (Z + Rt_inverse[2]) / RMU_inverse[2] * RMU_inverse -
    Rt_inverse

    return [wc[0,0], wc[1,1], Z]
```

$$\tilde{\mathbf{u}} = \mathbf{M}_{\text{int}} \mathbf{M}_{\text{ext}} \tilde{\mathbf{x}}_w \quad \longrightarrow \quad \tilde{\mathbf{x}}_w = \mathbf{M}_{\text{int}}^{-1} \mathbf{M}_{\text{ext}}^{-1} \tilde{\mathbf{u}}$$

3. Digital image processing | Probability-based segmentation

- Using ternary mask operations (**template matching**), we can identify objects by providing **a-priori knowledge about the shape of the object**

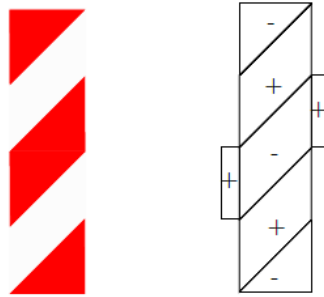


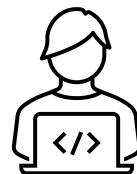
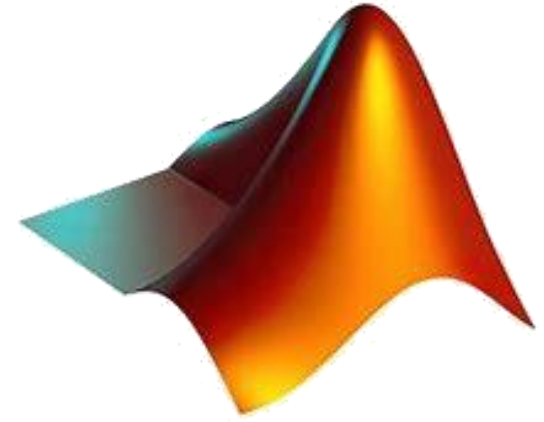
Fig. 22: A-priori object masks for template matching

- Using color model for initial segmentation
- Applying scale
binary mask regarding the estimated
inverse-perspective projection to distance
- Finding the optimal position of the mask through
its maximum response [Neu13]
- Estimation must be filtered with Kalman-Filter!

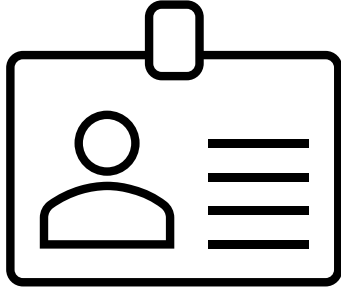


4. Software demonstration

1. Camera calibration
2. World2Pixel and Pixel2World projection
3. Road segmentation
4. Distance estimation on extracted objects
5. Color model generation
6. Color model application

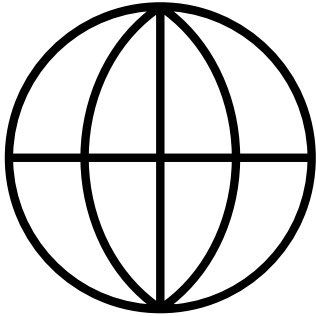


<https://github.com/schneider-daniel/ASAA>



Daniel Schneider

daniel.schneider@hs-Kempten.de



Assoc. with IFM – Institute for driver assistance and connected mobility, UAS Kempten, Germany

Interested on Master Thesis? Email me!