Self-Driving Cars

Lecture 9 – Reconstruction and Motion

Prof. Dr.-Ing. Andreas Geiger

Autonomous Vision Group

University of Tübingen / MPI-IS

EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

ellis
European Laboratory for Learning and Intelligent Systems

Agenda

9.1 Stereo Matching

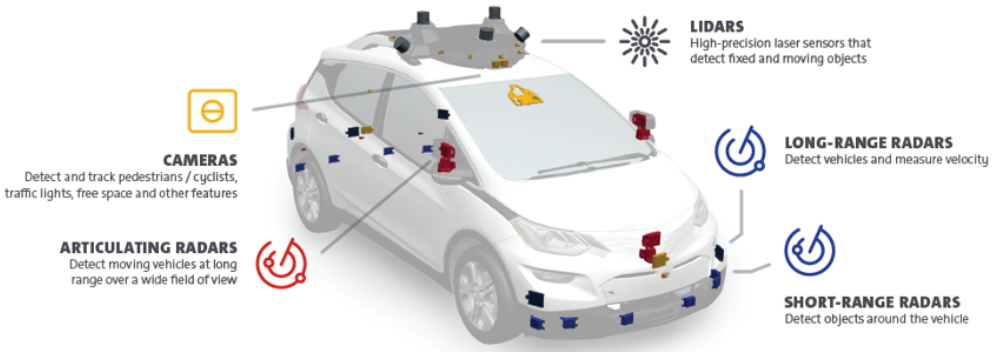
9.2 Freespace and Stixels

9.3 Optical Flow

9.4 Scene Flow

9.1
Stereo Matching

3D Vision



- Precise knowledge of the **environment's geometry** is crucial for safe navigation
- Self-driving vehicles typically combine **multiple different 3D sensor types**

https://www.allaboutcircuits.com/news/solid-state-lidar-faster-cheaper-better/

3D Vision

Active Sensors: (emit sound/radio/light waves)

- Ultrasonic sensors: short range (5 m)
  ⇒ Parking, blind spot detection
- Radar sensors: long range (300 m), low resolution
  ⇒ Adaptive cruise control (ACC)
- Lidar sensors: long range (100 m), mid resolution
  ⇒ Self-driving vehicle prototypes (expensive)

Passive Sensors: (don't emit any waves)

- Stereo cameras: mid range (50 m)
  ⇒ Cheap & high resolution, but require processing to obtain depth; accuracy depends on distance/texture

https://www.allaboutcircuits.com/news/solid-state-lidar-faster-cheaper-better/

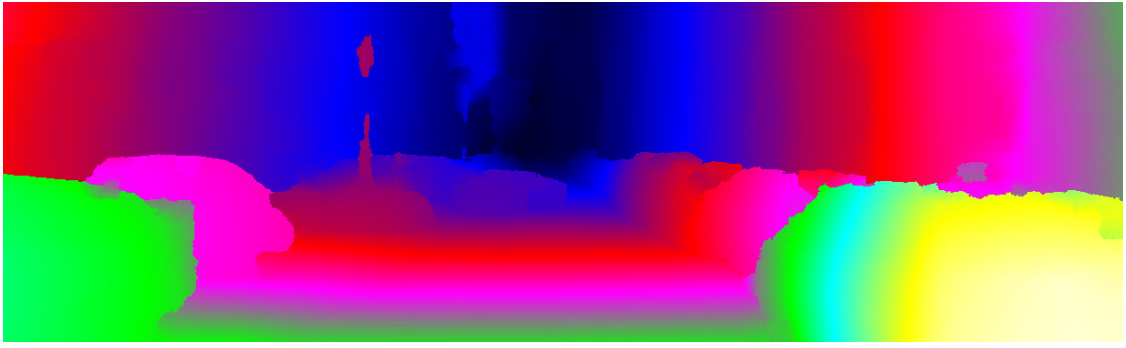
Disparity Estimation from a Stereo Image Pair

- **Input:** Images of 2 (laterally) displaced cameras captured at the same time
- **Output:** Horizontal displacement (=disparity) per pixel (“disparity map”)
- The **disparity** at a pixel is **anti-proportional** to its **scene depth** $\Rightarrow$ 2.5D depth map

Disparity Estimation from a Stereo Image Pair

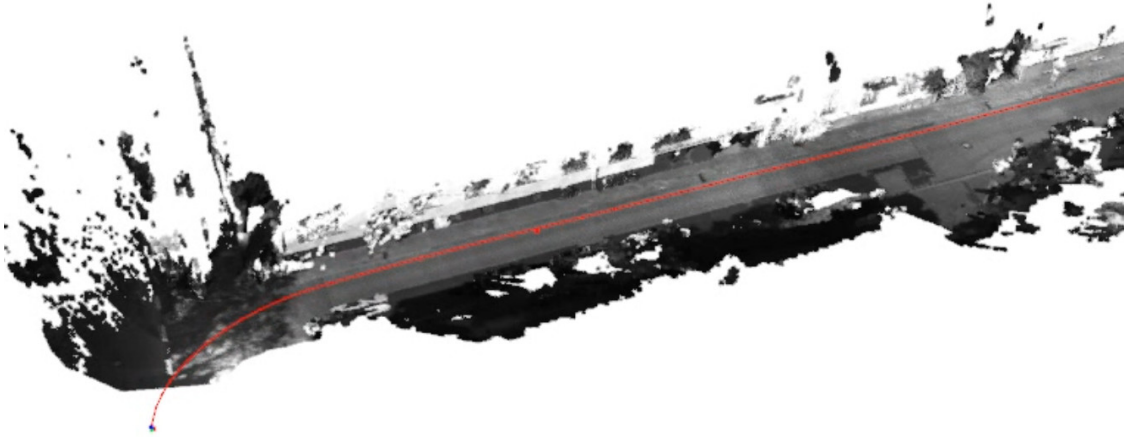
- **Input:** Images of 2 (laterally) displaced cameras captured at the same time
- **Output:** Horizontal displacement (=disparity) per pixel (“disparity map”)
- The **disparity** at a pixel is **anti-proportional** to its **scene depth** $\Rightarrow$ 2.5D depth map

Disparity Estimation from a Stereo Image Pair



- **Input:** Images of 2 (laterally) displaced cameras captured at the same time
- **Output:** Horizontal displacement (=disparity) per pixel (“disparity map”)
- The **disparity** at a pixel is **anti-proportional** to its **scene depth** ⇒ 2.5D depth map

Application: 3D Reconstruction



- Multiple 2.5D depth maps can be **fused** into large 3D reconstructions (e.g., via VO)

Geiger, Ziegler and Stiller: StereoScan: Dense 3d reconstruction in real-time. IV, 2011.

Binocular Stereo Matching

Task:

- Construct a **dense** 2.5D disparity map from 2 images of a static scene

Pipeline:

1. **Calibrate cameras** intrinsically and extrinsically
2. **Rectify images** given the calibration
3. **Compute disparity map** for reference image
4. **Remove outliers** using consistency/occlusion test
5. **Obtain depth** from disparity using triangulation
6. Optional: **Construct 3D model**, e.g., via volumetric fusing

Epipolar Geometry

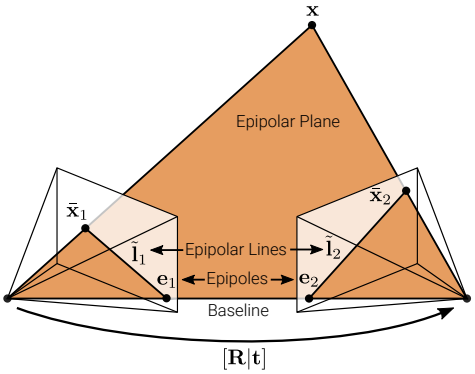
Epipolar Geometry

Epipolar Geometry:

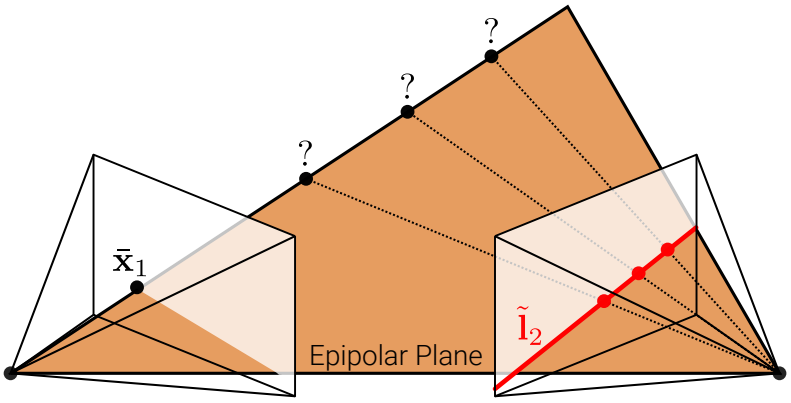
- 3D point $\mathbf{x}$ is projected to both images as $\bar{\mathbf{x}}_1 \propto \mathbf{K}_1 \mathbf{x}$ and $\bar{\mathbf{x}}_2 \propto \mathbf{K}_2 (\mathbf{R}\mathbf{x} + \mathbf{t})$
- Image correspondences are related by

$$\tilde{\mathbf{x}}_2^\top \tilde{\mathbf{E}} \tilde{\mathbf{x}}_1 = 0 \quad \text{with} \quad \tilde{\mathbf{x}}_i = \mathbf{K}_i^{-1} \bar{\mathbf{x}}_i$$

- The correspondence of pixel $\bar{\mathbf{x}}_1$ is located on the **epipolar line** $\tilde{\mathbf{l}}_2 = \tilde{\mathbf{E}} \tilde{\mathbf{x}}_1$
- The correspondence of pixel $\bar{\mathbf{x}}_2$ is located on the **epipolar line** $\tilde{\mathbf{l}}_1 = \tilde{\mathbf{E}}^\top \tilde{\mathbf{x}}_2$
- Epipolar lines intersect at the **epipoles**



Epipolar Geometry

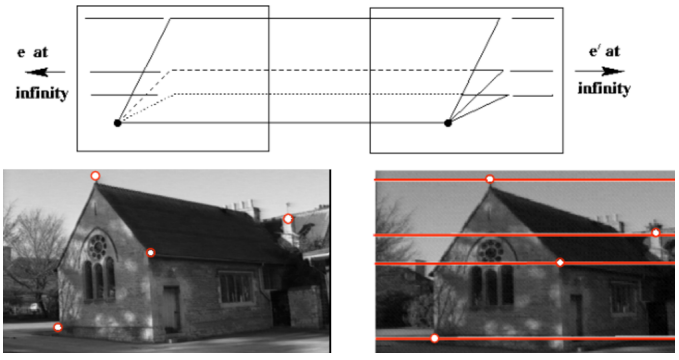


- A point $\bar{\mathbf{x}}_1$ in the left image must be located on the **epipolar line** $\tilde{\mathrm{l}}_2$
- This reduces correspondence search to a (much simpler) **1D problem**
- For VGA images: ∼640 instead of ∼300k hypotheses (factor 480 less)

Image Rectification

Image Rectification

What if both cameras face exactly the same direction?



- **Image planes** are **co-planar** ⇒ Epipoles at infinity, epipolar lines parallel
- Correspondences search along **horizontal scanlines** (simplifies implementation)
- However, in practice, it is impossible to perfectly align the two camera sensors

Image Rectification

What if the images are not in the required setup?

- **There is a trick:** We can **rewarp** them through **rotation,** mapping both image planes to a common plane parallel to the baseline, this is called **rectification**
- To rectify, we must only know $\mathbf{K}$ and the essential matrix $\tilde{\mathbf{E}}$ (see CV lecture 4.1)

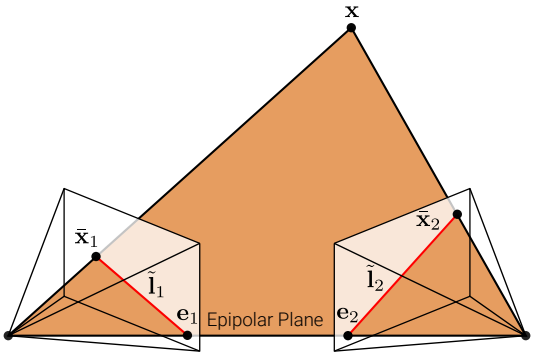
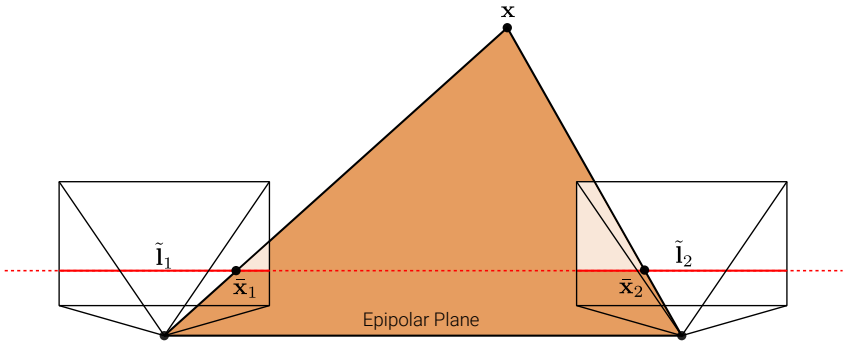


Image Rectification

What if the images are not in the required setup?

- **There is a trick:** We can **rewarp** them through **rotation,** mapping both image planes to a common plane parallel to the baseline, this is called **rectification**
- To rectify, we must only know $\mathbf{K}$ and the essential matrix $\tilde{\mathbf{E}}$ (see CV lecture 4.1)

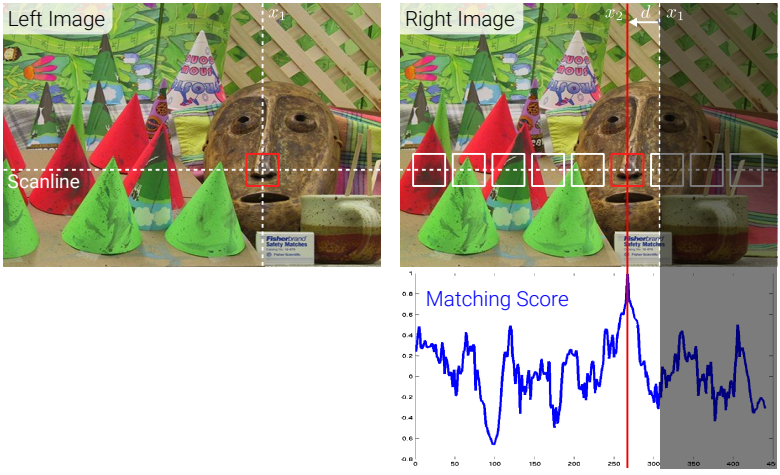


Rectification Example

- Correspondences are now located on the **same image row** as the query point

Stereo Matching

Similarity Metrics



- Determine correspondences through patch similarity (a single pixel is not enough)

Similarity Metrics

- Consider two $K \times K$ windows of pixels flattened to vectors $\mathbf{w}_L, \mathbf{w}_R \in \mathbb{R}^{K^2}$
- **Sum of squared differences (SSD):**

$$\text{SSD}(x, y, d) = \|\mathbf{w}_L(x, y) - \mathbf{w}_R(x - d, y)\|_2^2$$

- Numerous other similarity metrics exist (see, e.g., Szeliski, Chapter 12.3.)

Hirschmüller and Scharstein: Evaluation of Stereo Matching Costs on Images with Radiometric Differences. TPAMI, 2009.

Similarity Metrics

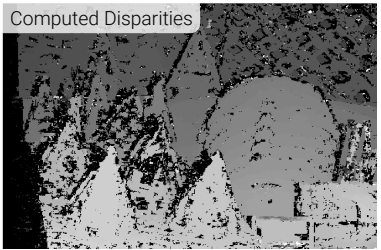
- Consider two $K \times K$ windows of pixels flattened to vectors $\mathbf{w}_L, \mathbf{w}_R \in \mathbb{R}^{K^2}$
- **Zero Normalized Cross-Correlation (ZNCC):** ($\bar{\mathbf{w}}$ denotes the mean of $\mathbf{w}$)

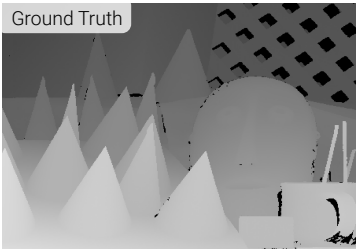
$$\text{ZNCC}(x, y, d) = \frac{\left(\mathbf{w}_L(x, y) - \bar{\mathbf{w}}_L(x, y)\right)^T \left(\mathbf{w}_R(x - d, y) - \bar{\mathbf{w}}_R(x - d, y)\right)}{\left\|\mathbf{w}_L(x, y) - \bar{\mathbf{w}}_L(x, y)\right\|_2 \left\|\mathbf{w}_R(x - d, y) - \bar{\mathbf{w}}_R(x - d, y)\right\|_2}$$

- Numerous other similarity metrics exist (see, e.g., Szeliski, Chapter 12.3.)

Hirschmüller and Scharstein: Evaluation of Stereo Matching Costs on Images with Radiometric Differences. TPAMI, 2009.

Block Matching

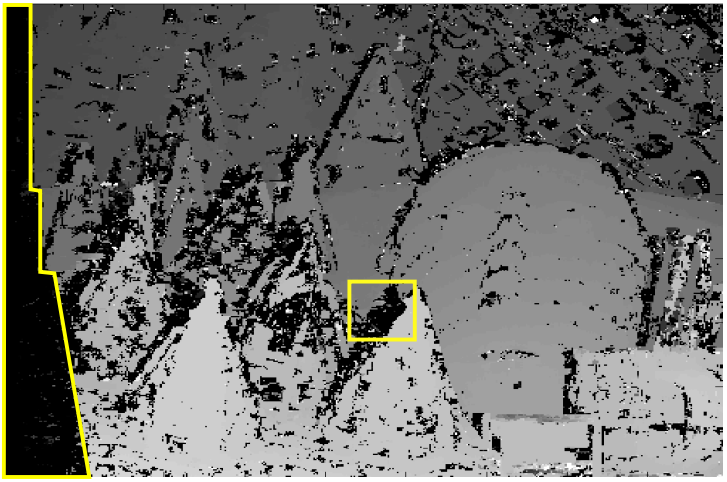




Block Matching:

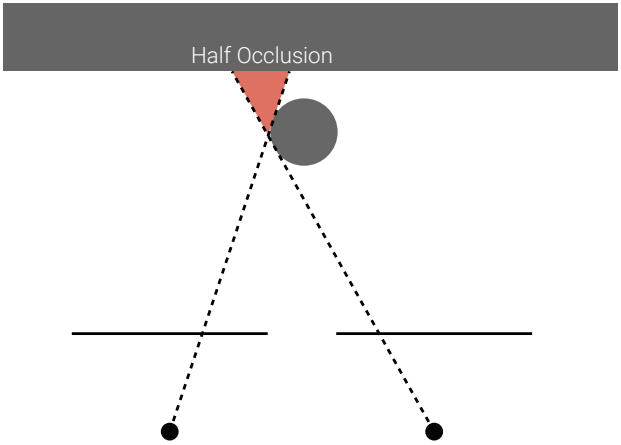
- Choose disparity range $[0, D]$
- For all pixels $\mathbf{x} = (x, y)$ compute the best disparity ("winner-takes-all" solution)
- Do this for both images, apply left-right consistency check to remove outliers

Block Matching: Half Occlusions



► Not all pixels can be matched as some are not visible in both cameras

Block Matching: Half Occlusions



- The red area is visible in the left image, but not in the right image

Block Matching: Assumption Violations

Assumption Violations:

- Block matching assumes that all pixels inside the window are displaced by d
- This is called the **fronto-parallel assumption** – but it is often invalid!
- Example 1: **Slanted surfaces** deform perspectively when the viewpoint changes

Block Matching: Assumption Violations

Assumption Violations:

- Block matching assumes that all pixels inside the window are displaced by d
- This is called the **fronto-parallel assumption** – but it is often invalid!
- Example 1: **Slanted surfaces** deform perspectively when the viewpoint changes

Block Matching: Assumption Violations

Assumption Violations:

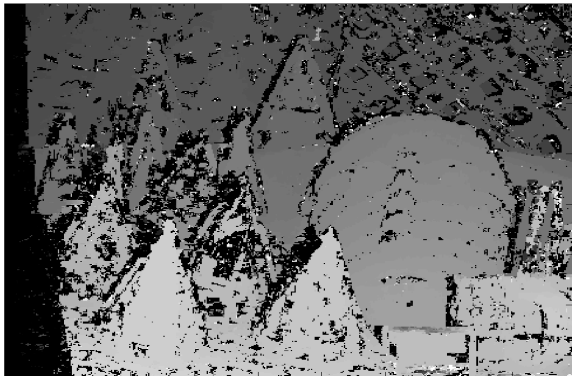
- Block matching assumes that all pixels inside the window are displaced by d
- This is called the **fronto-parallel assumption** – but it is often invalid!
- Example 2: The window content changes differently at **disparity discontinuities**

Block Matching: Assumption Violations

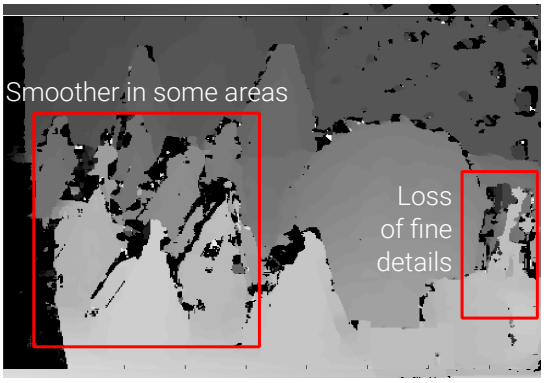
Assumption Violations:

- Block matching assumes that all pixels inside the window are displaced by d
- This is called the **fronto-parallel assumption** – but it is often invalid!
- Example 2: The window content changes differently at **disparity discontinuities**

Effect of Window Size



Window Size: 5x5 Pixels

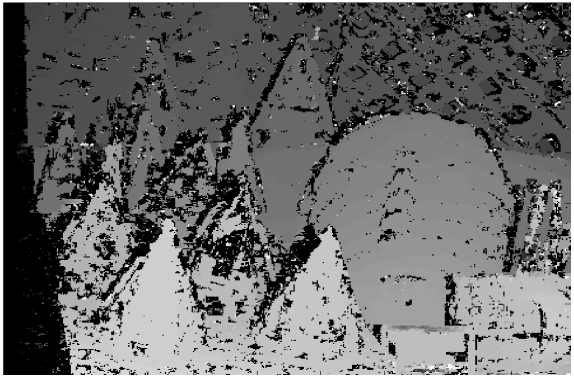


Window Size: 11x11 Pixels

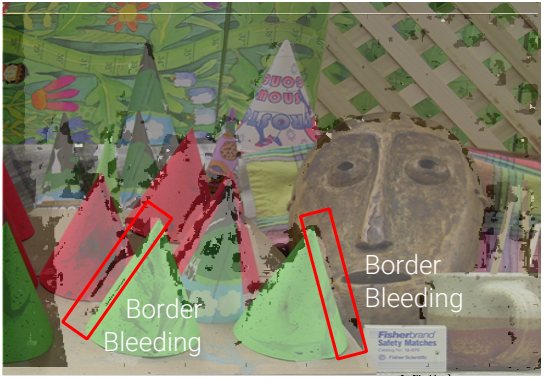
Tradeoff:

- Small windows lead to matching ambiguities and **noise** in the disparity maps
- Larger windows lead to smoother results, but **loss of detail** and border bleeding

Effect of Window Size



Window Size: 5x5 Pixels

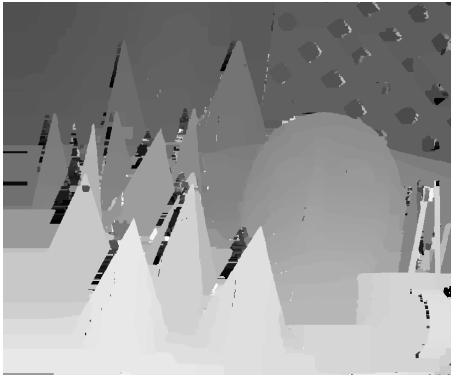


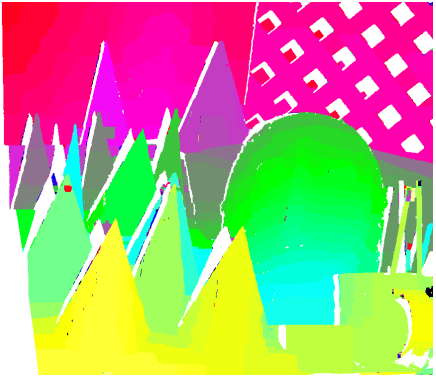
Window Size: 11x11 Pixels

Tradeoff:

- Small windows lead to matching ambiguities and **noise** in the disparity maps
- Larger windows lead to smoother results, but **loss of detail** and border bleeding

Left-Right Consistency Test





Left-Right Consistency Test:

- Outliers and half-occlusions can be detected via a left-right consistency test
- Compute disparity map for both images, verify if they map to each other (cycle)

Disparity to Depth

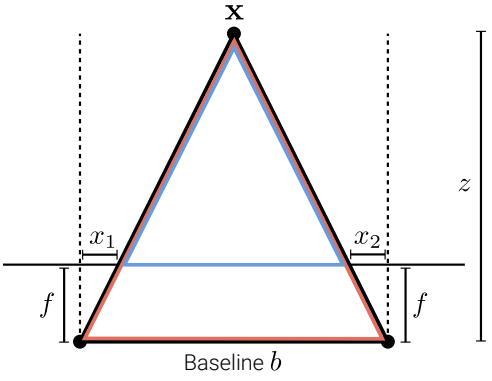
Disparity to Depth

How can we recover depth from disparity in the rectified case?

- The left and right ray must intersect as both lie on the same epipolar plane
- Let $x_1 > 0$, $x_2 < 0$ and $d = x_1 - x_2$. We have:

$$\frac{z-f}{b-d} = \frac{z}{b}$$
$$zb - fb = zb - zd$$
$$z = \frac{fb}{d}$$

- Remark: The depth error grows **quadratically** with depth (exercise)

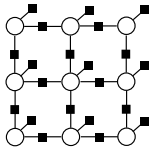


Spatial Regularization and Deep Learning

Spatial Regularization

- Incorporate **smoothness constraints** by specifying an energy function on a grid and **minimizing the energy** with respect to the **entire disparity map $\mathbf{D}$ jointly:**

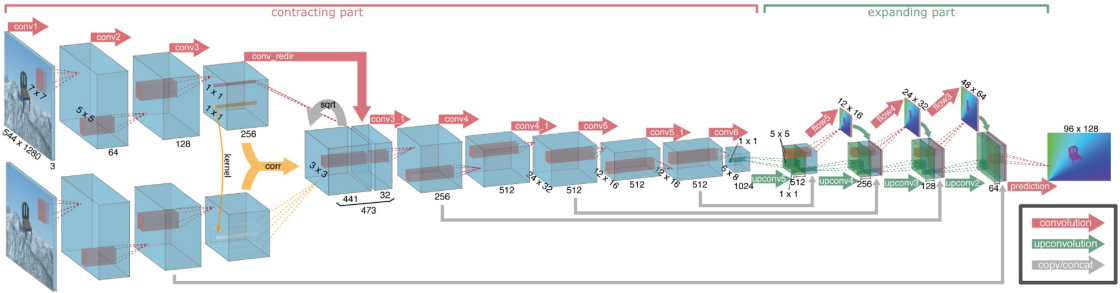
$$E(\mathbf{D}) = \sum_i \psi_{data}(d_i) + \lambda \sum_{i \sim j} \psi_{smooth}(d_i, d_j)$$



- $i \sim j$: neighboring pixels on a 4-connected grid
- **Unary** terms: Matching cost $\psi_{data}(d)$
- **Pairwise** terms: Smoothness between adjacent pixels, e.g.:
  - Potts: $\psi_{smooth}(d, d') = [d \neq d']$
  - Truncated l_1: $\psi_{smooth}(d, d') = \min(|d - d'|, \tau)$
- Various optimization techniques: belief propagation, graph cuts, .. (see CV lecture)

Huang, Lee and Mumford: Statistics of Range Images. CVPR, 2000.

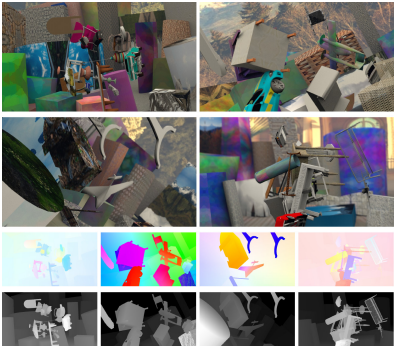
Deep Learning for Stereo Matching



- **DispNet** used a **U-Net** like architecture with skip-connections to retain details
- **Correlation layer** (40px displacement corresponding to 160px in input image)
- Multi-scale loss (disparity error in pixels), curriculum learning (learn easy-to-hard)

Mayer et al.: A Large Dataset to Train Convolutional Networks for Disparity, Optical Flow, and Scene Flow Estimation. CVPR, 2016.

Synthetic Datasets



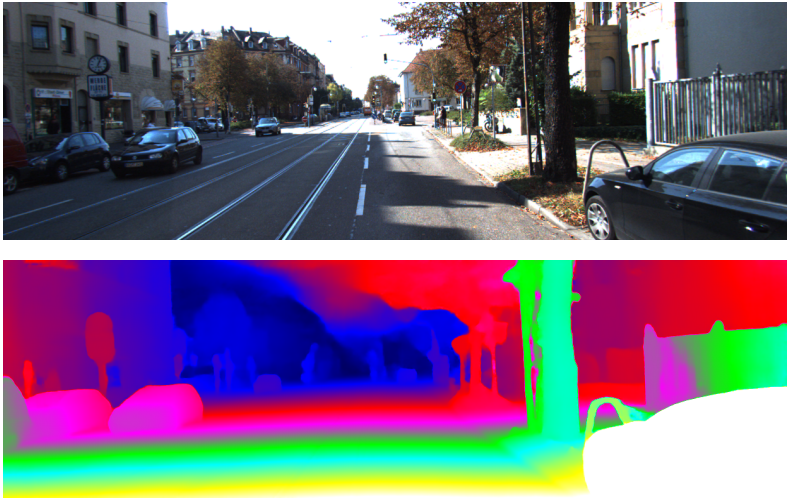
Flying Things

Monkaa

- **Pretraining** on large synthetic datasets (cheap annotations)
- **Finetuning** on little real data (expensive annotations)

Mayer et al.: A Large Dataset to Train Convolutional Networks for Disparity, Optical Flow, and Scene Flow Estimation. CVPR, 2016.

DispNet Results on KITTI Dataset



Mayer et al.: A Large Dataset to Train Convolutional Networks for Disparity, Optical Flow, and Scene Flow Estimation. CVPR, 2016.

9.2
Freespace and Stixels

Free Space Estimation

Problem Setting:

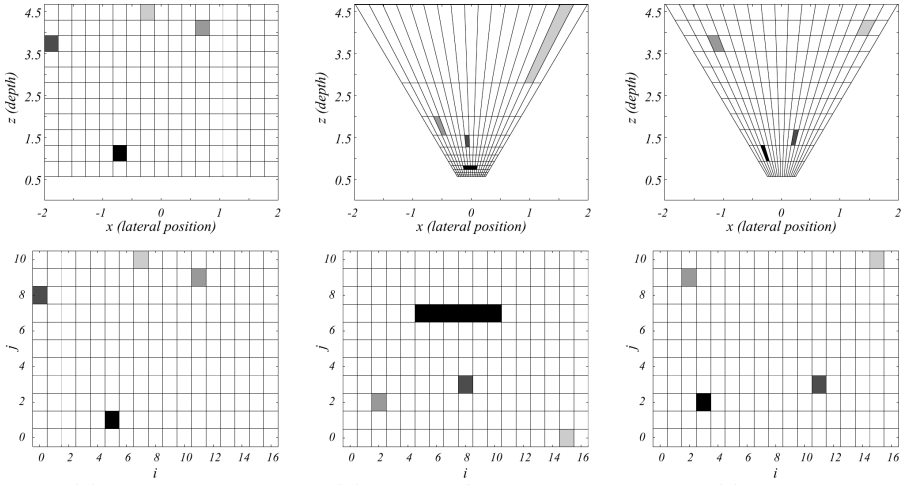
- Input: depth map per frame (e.g., from stereo reconstruction)
- Output: freespace in bird's eye view (BEV)
- Provides crucial information for (local) path-planning and collision avoidance

Approach:

- Integrate multiple disparity maps temporally (e.g., via VO)
- Convert depth measurements into BEV occupancy map
- Optimize freespace segmentation via energy minimization

Badino, Franke and Mester: Free Space Computation Using Stochastic Occupancy Grids and Dynamic Programming. ICCV Workshops, 2007.

Occupancy Grid Representations

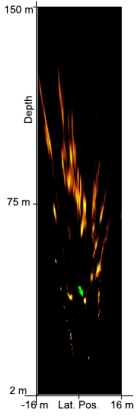


(a) Cartesian (b) Column/Disparity (c) Polar

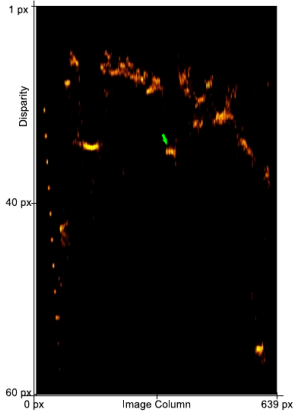
Badino, Franke and Mester: Free Space Computation Using Stochastic Occupancy Grids and Dynamic Programming. ICCV Workshops, 2007.

Occupancy Grid Representations

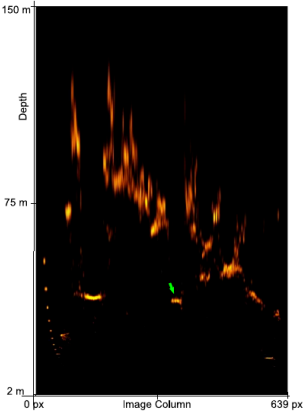
Image



Cartesian



Column/Disparity



Polar

Badino, Franke and Mester: Free Space Computation Using Stochastic Occupancy Grids and Dynamic Programming. ICCV Workshops, 2007.

Occupancy Map

Depth-to-Occupancy Conversion: (polar representation: image column x / depth z)
Let (x_i, z_i) denote the x-coordinate and depth value ($z = fb/d$) of pixel $i \in \{1, \ldots, N\}$.
We estimate the occupancy likelihood at location (x, z) in the occupancy map using:

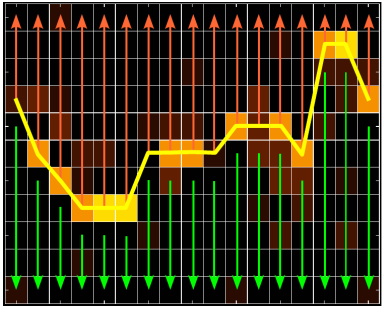
$$O(x, z) = \sum_{i=1}^{N} \exp\left(-\gamma_x \left(x - x_i\right)^2 - \gamma_z \left(z - z_i\right)^2\right)$$

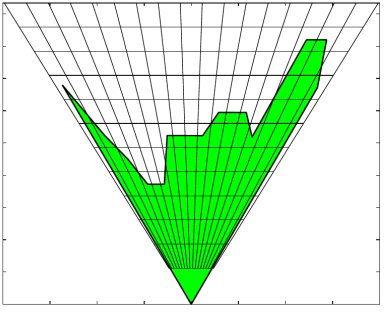
This kernel estimator accumulates evidence of nearby observations.
Badino et al. define a data term for each image column $j \in \{1, \ldots, M\}$ as follows:

$$\psi_z(z_j) = \frac{1}{O(x_j, z_j)}$$

Badino, Franke and Mester: Free Space Computation Using Stochastic Occupancy Grids and Dynamic Programming. ICCV Workshops, 2007.

Optimization



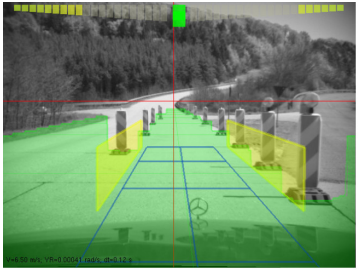


Optimization Problem: (solved via dynamic programming)

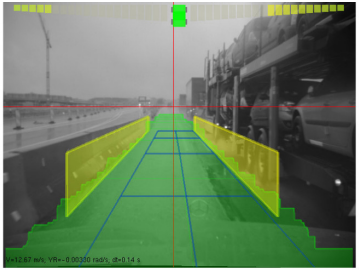
$$E(z_1, \ldots, z_M) = \sum_j \psi_z(z_j) + \lambda \sum_j \psi_s(z_j, z_{j+1}) \quad \text{with} \quad \psi_s(z_j, z_{j+1}) = \min(|z_j - z_{j+1}|, \tau)$$

- z_j = distance to obstacle at column j, ψ_z = data term, ψ_s = smoothness term

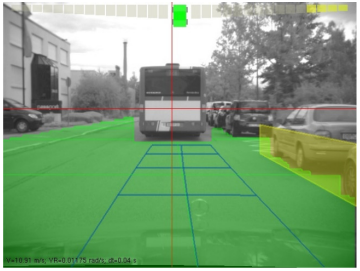
Results



(a) Highway.



(b) Freeway.

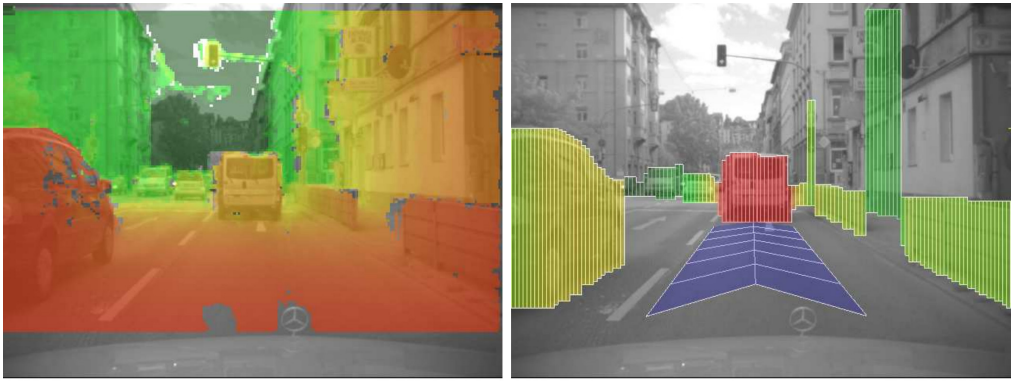


(c) Downtown.

- Height of obstacles unknown / only freespace estimation

Badino, Franke and Mester: Free Space Computation Using Stochastic Occupancy Grids and Dynamic Programming. ICCV Workshops, 2007.

The Stixel World

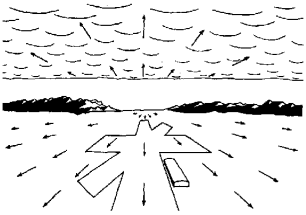


(a) Dense disparity image (SGM result) (b) Stixel representation

- "Stixel World" estimates multiple stixels per image column (i.e., segmentation)

9.3
Optical Flow

Optical Flow



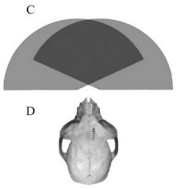
- Optical flow is the **apparent motion** of objects, surfaces, and edges in a visual scene caused by the relative motion between an observer and a scene.
- Knowing the past/current motion allows to make predictions about the future
- For example, we might predict the location of a vehicle 1 second into the future
- However, these predictions are in image space, not in 3D space ($\Rightarrow$ scene flow)

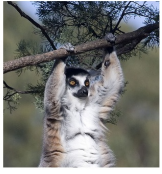
J. J. Gibson, 1950: The Ecological Approach to Visual Perception

Stereo vs. Optical Flow

Stereo

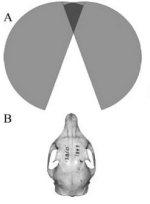
- 2 images at the same time
- Only camera motion
- 1D estimation problem
- Monkeys



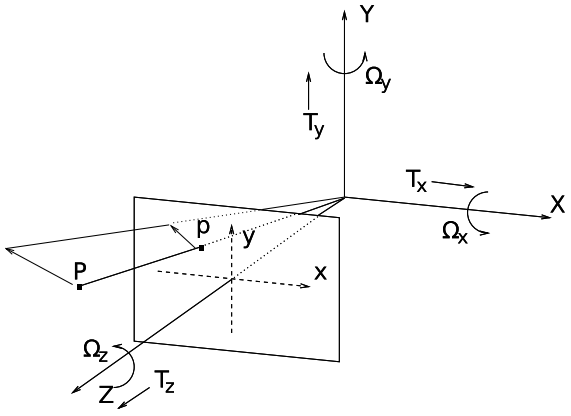


Optical Flow

- 2 images at 2 time steps
- Camera and object motion
- 2D estimation problem
- Squirrels



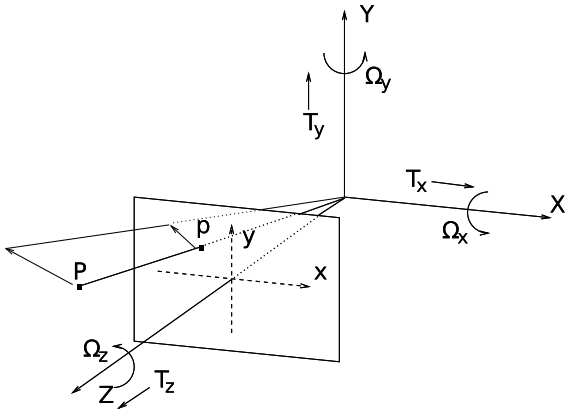
Optical Flow



Motion field:

- 2D motion field representing the **projection of the 3D motion** of points in the scene onto the image plane
- Can be the result of camera motion or object motion (or both)

Optical Flow



Optical flow:

- 2D velocity field describing the **apparent motion** in the image (i.e., the displacement of pixels looking “similar”)
- Optical flow $\neq$ motion field! Why?

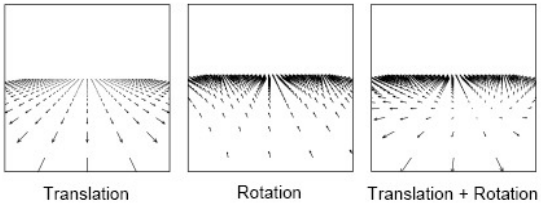
Thought Experiment

- Lambertian ball
  rotating in 3D
- What does the 2D
  motion field look like?
- What does the 2D
  optical flow field look like?

Thought Experiment

- Stationary specular ball
  moving light source
- What does the 2D
  motion field look like?
- What does the 2D
  optical flow field look like?

Optical Flow Field



The optical flow fields tell us something (maybe ambiguous) about:

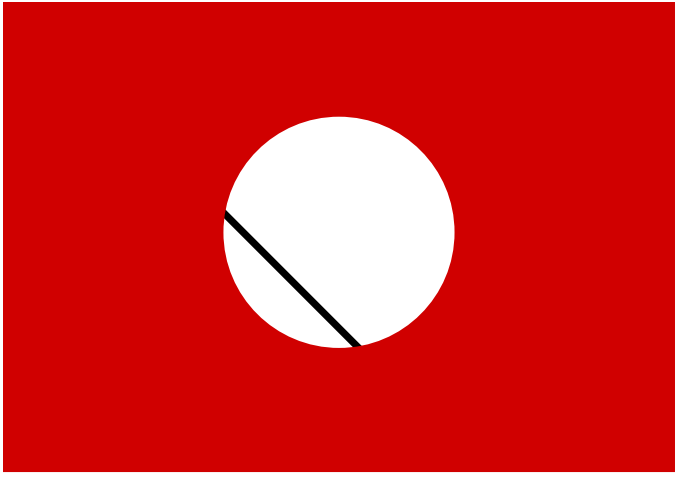
- The **3D structure** of the world
- The **motion of objects** in the viewing area
- The **motion of the observer** (if any)

In contrast to stereo:

- No epipolar geometry $\Rightarrow$ **2D estimation problem** $\Rightarrow$ aperture problem

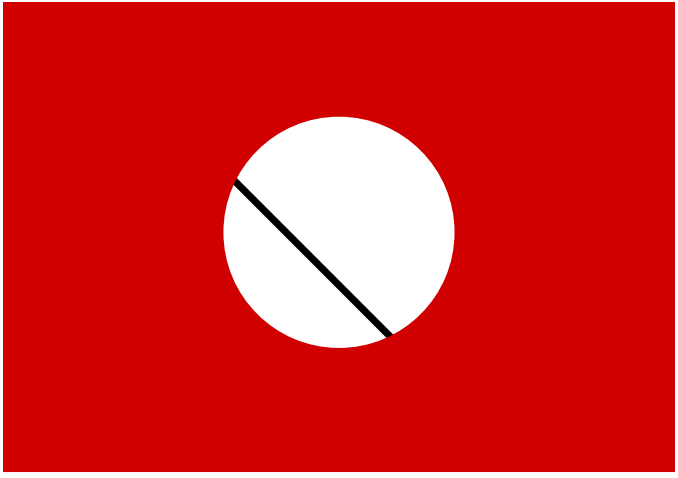
Aperture Problem

In which direction does the line move?



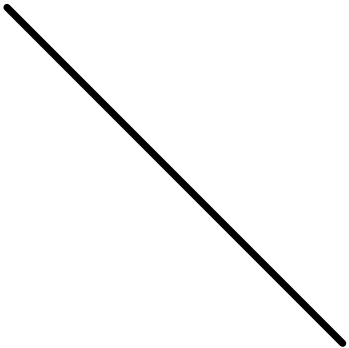
Aperture Problem

In which direction does the line move?



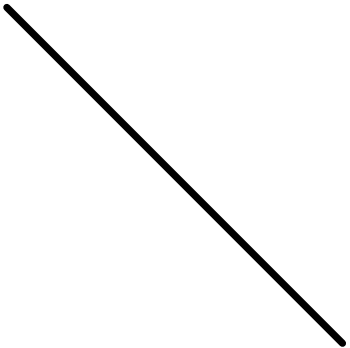
Aperture Problem

Now the full picture ...



Aperture Problem

Now the full picture …



Aperture Problem

- Aperture problem: A single observation is not enough to determine flow
- Barber Pole: What is the motion field? What is the optic flow field?

Determining Optical Flow

Horn-Schunck Optical Flow

- Consider the image I as a function of continuous variables x, y, t
- Consider $u(x, y)$ and $v(x, y)$ as continuous flow fields (functions)
- Minimize the following **energy functional**

$$
\begin{aligned}
E(u, v) &= \iint \underbrace{\left(I(x + u(x, y), y + v(x, y), t + 1) - I(x, y, t)\right)^2}_{\text{quadratic penalty for brightness change}} \\
&+ \lambda \cdot \underbrace{\left(\|\nabla u(x, y)\|^2 + \|\nabla v(x, y)\|^2\right)}_{\text{quadratic penalty for flow change}} \, dx \, dy
\end{aligned}
$$

with regularization parameter λ and gradient $\nabla = \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}\right)$.

Horn & Schunck: Determining Optical Flow. Artificial Intelligence, 1981.

Horn-Schunck Optical Flow

$$
\begin{aligned}
E(u,v) &= \iint \left(I(x+u(x,y), y+v(x,y), t+1) - I(x,y,t)\right)^2 \\
&+ \lambda \cdot \left(\|\nabla u(x,y)\|^2 + \|\nabla v(x,y)\|^2 \right) dx\, dy
\end{aligned}
$$

- Minimizing this directly is a hard problem because the energy is highly **non-convex** and has many local optima
- Solution: **linearize** the brightness constancy assumption

Horn & Schunck: Determining Optical Flow. Artificial Intelligence, 1981.

Horn-Schunck Optical Flow

First-Order Multivariable Taylor Series:

$$f(x,y) \overset{a,b}{\approx} f(a,b) + \frac{\partial f(a,b)}{\partial x}(x-a) + \frac{\partial f(a,b)}{\partial y}(y-b)$$

Therefore, we have:

$$\begin{aligned} & I(x+u(x,y), y+v(x,y), t+1) \\ \overset{x,y,t}{\approx}\ & I(x,y,t) + I_x(x,y,t)(x+u(x,y)-x) \\ & + I_y(x,y,t)(y+v(x,y)-y) + I_t(x,y,t)(t+1-t) \\ =\ & I(x,y,t) + I_x(x,y,t)\,u(x,y) + I_y(x,y,t)\,v(x,y) + I_t(x,y,t) \end{aligned}$$

Horn & Schunck: Determining Optical Flow. Artificial Intelligence, 1981.

Horn-Schunck Optical Flow

Thus, the optical flow energy functional

$$\begin{aligned} E(u,v) &= \iint \left(I(x+u(x,y), y+v(x,y), t+1) - I(x,y,t)\right)^2 \\ &+ \lambda \cdot \left(\|\nabla u(x,y)\|^2 + \|\nabla v(x,y)\|^2\right) dx\, dy \end{aligned}$$

is approximated by the following **linearized equation:**

$$\begin{aligned} E(u,v) &= \iint \left(I_x(x,y,t)u(x,y) + I_y(x,y,t)v(x,y) + I_t(x,y,t)\right)^2 \\ &+ \lambda \cdot \left(\|\nabla u(x,y)\|^2 + \|\nabla v(x,y)\|^2\right) dx\, dy \end{aligned}$$

Horn & Schunck: Determining Optical Flow. Artificial Intelligence, 1981.

Horn-Schunck Optical Flow

Spatial discretization of the equation

$$\begin{aligned} E(u,v) &= \iint \left(I_x(x,y,t)u(x,y) + I_y(x,y,t)v(x,y) + I_t(x,y,t)\right)^2 \\ &+ \lambda \cdot \left(\|\nabla u(x,y)\|^2 + \|\nabla v(x,y)\|^2\right) dx\, dy \end{aligned}$$

leads to the following **discretized objective:**

$$\begin{aligned} E(\mathbf{U},\mathbf{V}) &= \sum_{x,y} \left(I_x(x,y)\, u_{x,y} + I_y(x,y)\, v_{x,y} + I_t(x,y)\right)^2 \\ &+ \lambda \cdot \left((u_{x,y} - u_{x+1,y})^2 + (u_{x,y} - u_{x,y+1})^2 + (v_{x,y} - v_{x+1,y})^2 + (v_{x,y} - v_{x,y+1})^2\right) \end{aligned}$$

This objective is quadratic in the flow maps $\mathbf{U}, \mathbf{V}$ and thus has a unique optimum.

Horn & Schunck: Determining Optical Flow. Artificial Intelligence, 1981.

Horn-Schunck Optical Flow

$$
\begin{aligned}
E(\mathbf{U},\mathbf{V}) &= \sum_{x,y} \left(I_x(x,y)\,u_{x,y} + I_y(x,y)\,v_{x,y} + I_t(x,y)\right)^2 \\
&+ \lambda \cdot \big((u_{x,y} - u_{x+1,y})^2 + (u_{x,y} - u_{x,y+1})^2 + \\
&\qquad (v_{x,y} - v_{x+1,y})^2 + (v_{x,y} - v_{x,y+1})^2\big)
\end{aligned}
$$

- Differentiate wrt. $\mathbf{U}, \mathbf{V}$ and set the gradient to 0
- Results in a huge but **sparse linear system**
- Can be solved using **standard techniques** (e.g., Gauss-Seidel, SOR)
- However, linearization works only for **small motions**
- Solution: Iterative & coarse-to-fine estimation

Horn & Schunck: Determining Optical Flow. Artificial Intelligence, 1981.

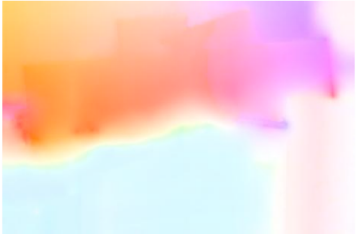
Optical Flow Visualization

Visualization of Results:

- An optical flow field (estimate or gt) provides two values per pixel: flow in u- and v- direction
- How to visualize 2 values per pixel?
- A dense arrow grid is inconvenient
- Flow direction encoded as color
- Flow magnitude encoded as intensity

Horn & Schunck: Determining Optical Flow. Artificial Intelligence, 1981.

Results of Horn & Schunck

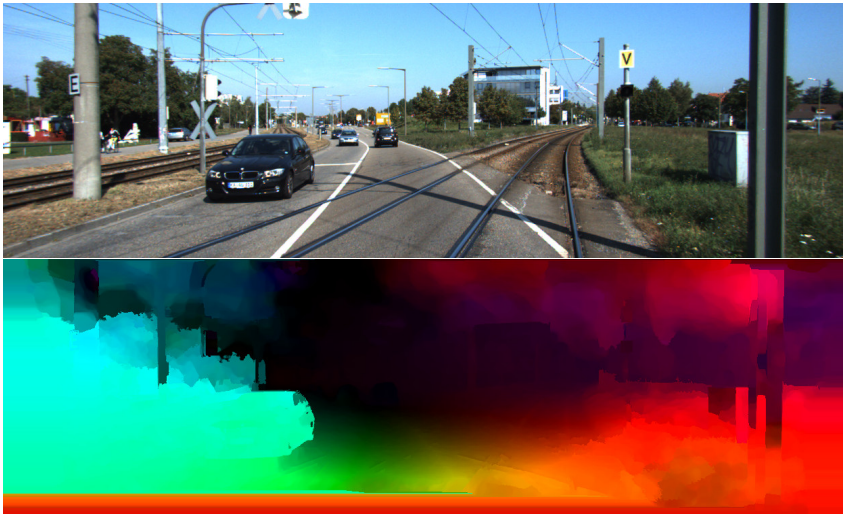


Horn & Schunck: Determining Optical Flow. Artificial Intelligence, 1981.

Results of Horn & Schunck with Robust Penalties

Sun, Roth and Black: Secrets of optical flow estimation and their principles. CVPR, 2010.

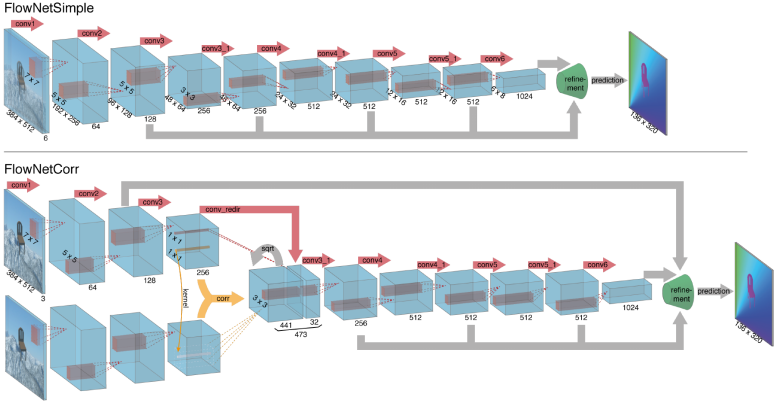
Results of Horn & Schunck with Robust Penalties



Sun, Roth and Black: Secrets of optical flow estimation and their principles. CVPR, 2010.

Deep Learning for Optical Flow Estimation

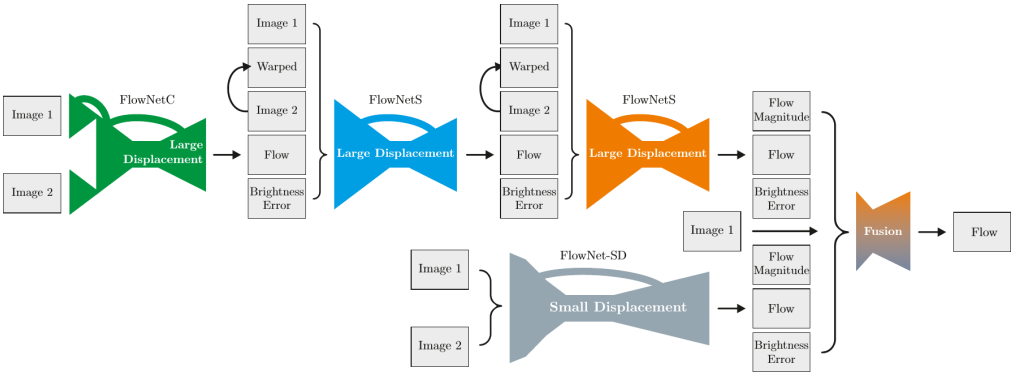
FlowNet



- Encoder: Convolution with stride, decoder: Upconvolution, skip-connections
- Multi-scale loss (EPE in pixels), curriculum learning, synth. training data

Dosovitskiy et al.: FlowNet: Learning Optical Flow,with Convolutional Networks. ICCV, 2015.

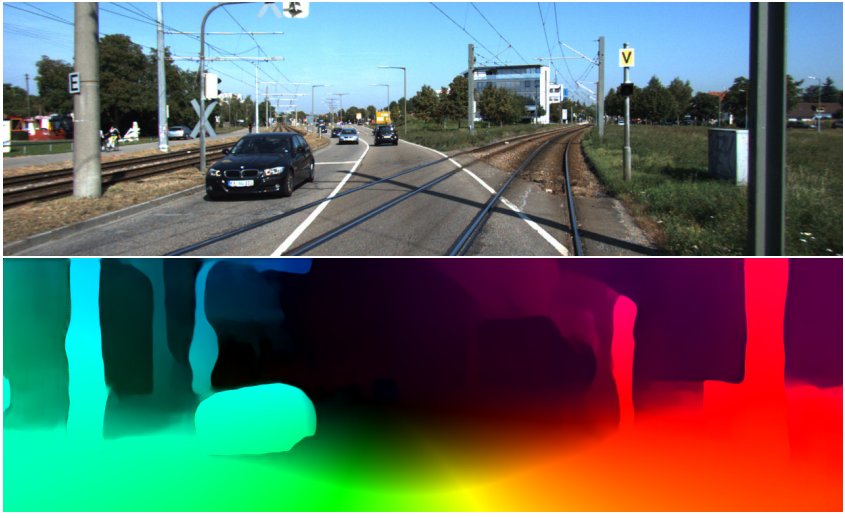
FlowNet2



[Ilg et al., CVPR 2017]

- Combination (stacking) of multiple FlowNets with warping

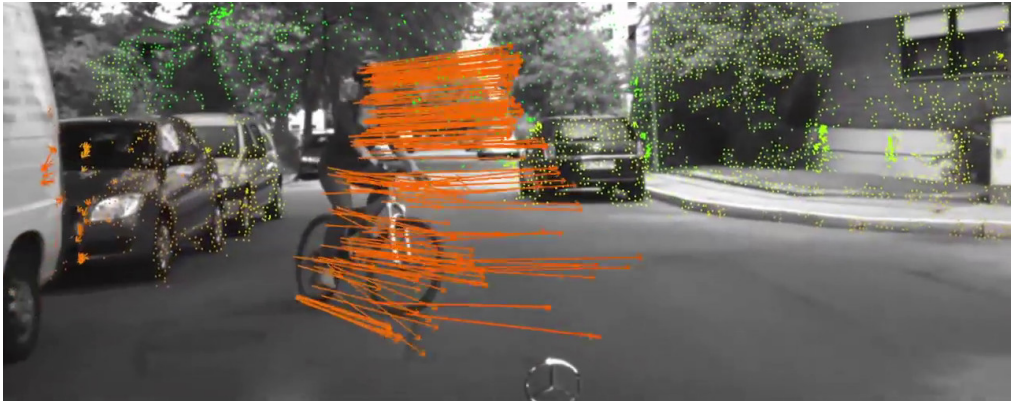
FlowNet2 Results on KITTI



Ilg, Mayer, Saikia, Keuper, Dosovitskiy and Brox: FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks. CVPR, 2017.

9.4
Scene Flow

Scene Flow Estimation



- In practice, **2D motion** estimation (in the image domain) is often not sufficient
- We live in 3D space, hence we must estimate/predict **motion in 3D** $\Rightarrow$ scene flow

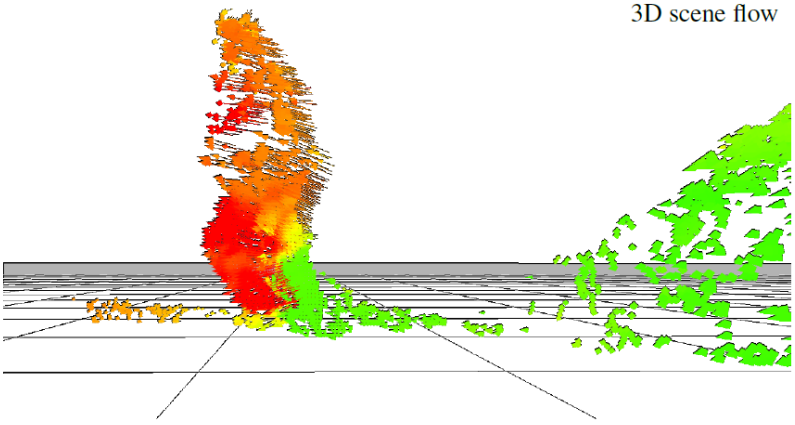
Franke, Rabe, Badino and Gehrig: 6D-Vision - Fusion of Motion and Stereo for Robust Environment Perception. GCPR, 2015.

3D Scene Flow

"Scene flow is a dense three-dimensional vector field defined for each point on every surface in the scene" [Vedula et al. 2005]

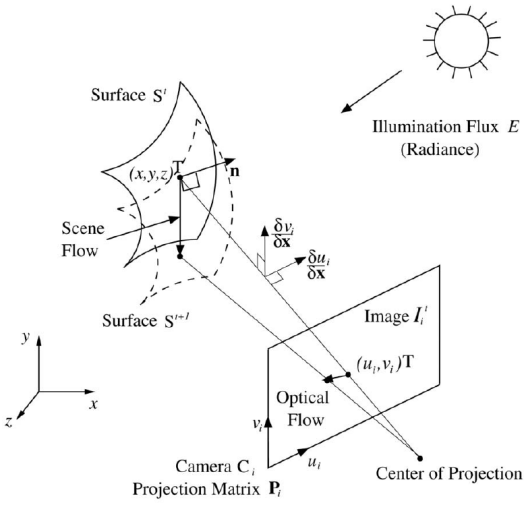
left image at time t

left image at time $t+1$



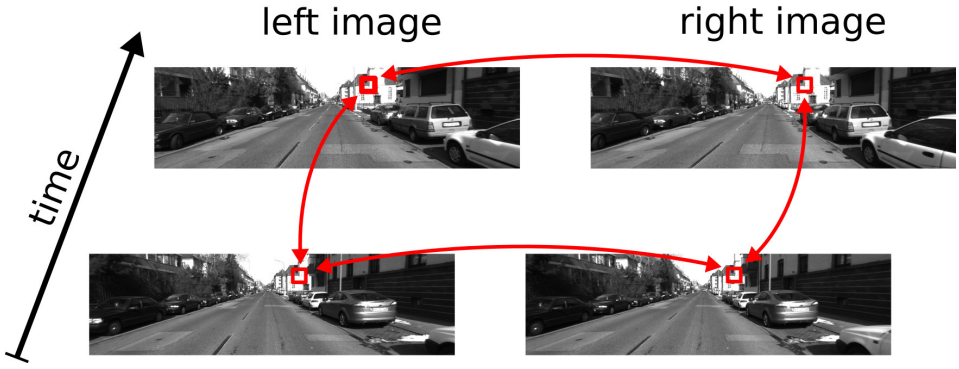
3D Scene Flow

- The scene flow is the **instantaneous 3D motion** of every point $(x, y, z)^\top$ on every surface in the scene
- Optical flow is the 2D projection of the scene flow into an image
- Vedula et al. compute 3D scene flow from 2D optical flow estimates
- Later works estimate scene flow directly from images using the brightness constancy assumption
- How many observations required?



Vedula, Baker, Rander, Collins and Kanade: Three-Dimensional Scene Flow. TPAMI, 2005.

3D Scene Flow

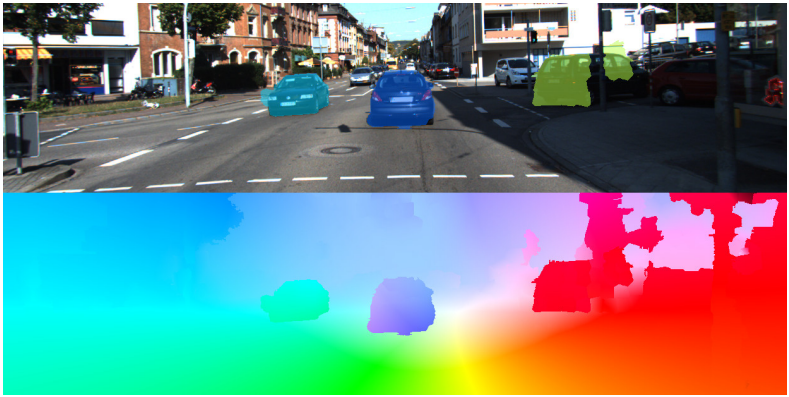


- To estimate scene flow, we require at least **4 images** or 2 images with depth
- Hence, scene flow effectively **combines the problems of stereo and optical flow**
- Common representation: Pixel location, optical flow, disparity in 1st+2nd frame

Vedula, Baker, Rander, Collins and Kanade: Three-Dimensional Scene Flow. TPAMI, 2005.

3D Scene Flow

- Naïve approach: Combine separately computed optical flow and stereo results
- Better approach: Reason jointly about flow, stereo, segmentation, rigid motions:



Menze and Geiger: Object scene flow for autonomous vehicles. CVPR, 2015.

3D Scene Flow

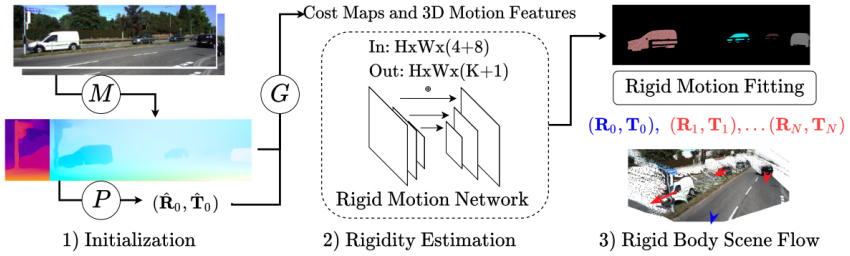


Figure 4: We detect and estimate rigid motions in three steps: First, depth and optical flow are computed using off-the-shelf networks (M) and camera motion is estimated by epipolar geometry (P) given two frames. Then, rigidity cost maps and rectified scene flow are computed (G) and fed into a two-stream network that produces the segmentation masks of a rigid background and an arbitrary number of rigidly moving instances. Finally, we fit rigid transformations for the background and each rigid instance to update their depth and 3D scene flow.

- Recurring idea: Exploit fact that scene is composed of **rigidly moving objects**
- Currently ranked 1st on KITTI Scene Flow Benchmark

  `http://www.cvlibs.net/datasets/kitti/eval_scene_flow.php`

Summary

- Stereo uses 2 images captured at the same time to infer depth
- Compared to other sensors, stereo vision is cheap and passive
- However, stereo requires processing and accuracy depends on depth/texture
- When using block matching, we face a trade-off wrt. window size
- Freespace approaches convert depth maps into BEV occupancy maps
- Occupancy maps are useful for path planning and collision avoidance
- Optical flow estimates the apparent motion in the image domain
- In contrast to stereo, optical flow cannot exploit epipolar geometry
- Instead, we must infer 2 variables per pixel $\Rightarrow$ ill-posed
- Scene flow estimates 3D motion, hence combining stereo and optical flow