Flow Estimation

Min Bai

University of Toronto

February 8, 2016

Outline

- Optical Flow - Continued

Outline

- Optical Flow - Continued
  - Better descriptors for matching - DeepFlow (ICCV 2013)

Outline

- Optical Flow - Continued
  - Better descriptors for matching - DeepFlow (ICCV 2013)
  - Better dense optical flow generation - EpicFlow (CVPR 2015)

Outline

- Optical Flow - Continued
    - Better descriptors for matching - DeepFlow (ICCV 2013)
    - Better dense optical flow generation - EpicFlow (CVPR 2015)
- Scene Flow

Outline

- Optical Flow - Continued
  - Better descriptors for matching - DeepFlow (ICCV 2013)
  - Better dense optical flow generation - EpicFlow (CVPR 2015)
- Scene Flow

Outline

- Optical Flow - Continued
  - Better descriptors for matching - DeepFlow (ICCV 2013)
  - Better dense optical flow generation - EpicFlow (CVPR 2015)
- Scene Flow
  - Optic Flow vs Scene Flow

Outline

- Optical Flow - Continued
    - Better descriptors for matching - DeepFlow (ICCV 2013)
    - Better dense optical flow generation - EpicFlow (CVPR 2015)
- Scene Flow
    - Optic Flow vs Scene Flow
    - 3D to 2D Projection Review

Outline

- Optical Flow - Continued
  - Better descriptors for matching - DeepFlow (ICCV 2013)
  - Better dense optical flow generation - EpicFlow (CVPR 2015)
- Scene Flow
  - Optic Flow vs Scene Flow
  - 3D to 2D Projection Review
  - Epipolar Constraint Review

Outline

- Optical Flow - Continued
  - Better descriptors for matching - DeepFlow (ICCV 2013)
  - Better dense optical flow generation - EpicFlow (CVPR 2015)
- Scene Flow
  - Optic Flow vs Scene Flow
  - 3D to 2D Projection Review
  - Epipolar Constraint Review
  - Static scene structure from monocular vision - Robust Monocular Epipolar Flow Estimation (CVPR13)

Outline

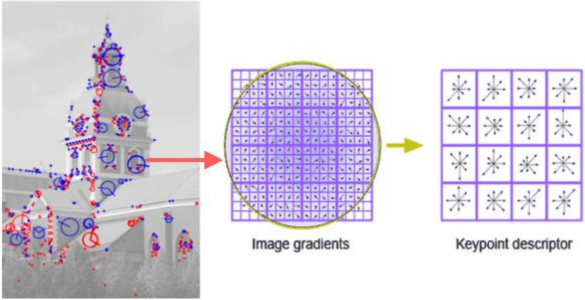
- Optical Flow - Continued
  - Better descriptors for matching - DeepFlow (ICCV 2013)
  - Better dense optical flow generation - EpicFlow (CVPR 2015)
- Scene Flow
  - Optic Flow vs Scene Flow
  - 3D to 2D Projection Review
  - Epipolar Constraint Review
  - Static scene structure from monocular vision - Robust Monocular Epipolar Flow Estimation (CVPR13)
  - Dynamic scene structure from stereo + time sequence - 3D Scene Flow Estimation with a Piecewise Rigid Scene Model (CVPR15)

DeepFlow - Motivation

- DeepFlow: Large Displacement Optical Flow with Deep Matching
  - Weinzaepfel et al, ICCV 2013
- Challenges in finding correct matching points in image pairs:
  - Distortions due to perspective changes
  - Distortions due to large displacements of object between image captures
  - Occlusion, scale, texture, etc
- DeepFlow addresses first two items

DeepFlow - Comparison with SIFT

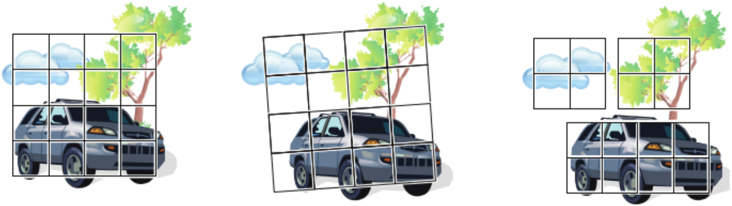
- Creates descriptive feature unique to one point based on context patch



Source: *Bandara @ codeproject.com*

DeepFlow - Comparison with SIFT

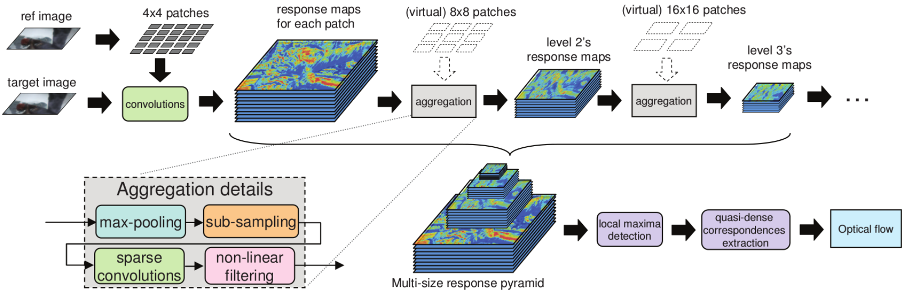
- Left and middle - traditional SIFT matching
  - Based on rigid patch
- Right - more optimal matching
  - Non rigid, deformable patch



Source: *Weinzaepfel et al, ICCV 2013*

DeepFlow - Architecture

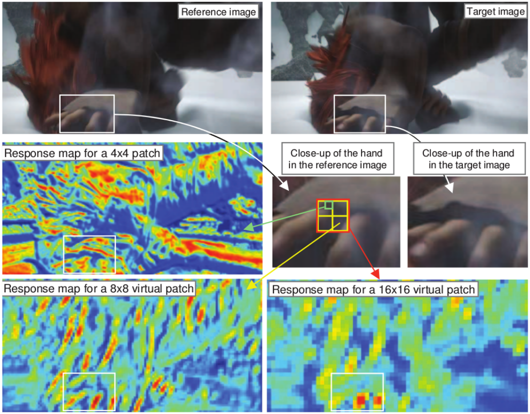
- Inspired by deep convolutional neural networks



Source: *Weinzaepfel et al, ICCV 2013*

DeepFlow - Architecture

- Progression of image data through architecture



Source: *Weinzaepfel et al, ICCV 2013*

DeepFlow - Energy optimization

- Traditionally, flow field is estimated by minimizing the following energy over the image:

$$E(w) = E_{\text{data}} + \alpha E_{\text{smoothness}}$$

  - data term encourages color and gradient consistency
  - smoothness term discourages abrupt changes in flow field
- DeepFlow adds one more term to encourage flow estimation to equal displacements from matching

$$E(w) = E_{\text{data}} + \alpha E_{\text{smoothness}} + \beta E_{\text{matching}}$$

  - matching term also takes quality of matches into account

EpicFlow

- EpicFlow: Edge-Preserving Interpolation of Correspondences for Optical Flow
  - Revaud et al, CVPR2015
- Based in part on Deep Matching (as developed in DeepFlow)
- Also incorporates knowledge about visual edges in images

EpicFlow - Motivation

- Commonly used method - coarse-to-fine interpolation / energy minimization
  - Start with smoothed or down-sampled image, estimate flow
  - Use estimated flow for a less down-sampled image for further refinement

$$E_{\text{smooth}} = \Psi(\|\nabla u\|^2 + \|\nabla v\|^2)$$

- Ψ is a robust penalization function

EpicFlow - Motivation

- Question:
  - Should smoothness in flow field be uniformly encouraged?
  - What effect does this have on natural motion discontinuities in 3D?
  - What effect does this have on small but fast moving objects?

Source: *KITTI*

EpicFlow - Motivation

- Question:
    - Should smoothness in flow field be uniformly encouraged?
    - What effect does this have on natural motion discontinuities in 3D?
    - What effect does this have on small but fast moving objects?

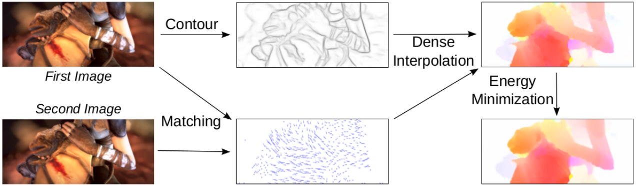
Source: *KITTI*

EpicFlow - Key Insight

- Motion discontinuities tend to occur at image edges
  - In the natural world, the same objects tend to be coherent in color (compared to background)
  - Abrupt changes in true optical flow tends to be due to 3D points not belonging to the same rigidly moving body or abrupt changes in depth

EpicFlow - Architecture

- Run edge detection algorithm (based on Structured Forests for Fast Edge Detection, ICCV 2013)
- Run Deep Matching (same algorithm as discussed previously)
- Flow field interpolation through edge-aware distances



  - This skips the coarse-to-fine pyramid!
- One level energy minimization

Source: *Revaud et al, CVPR 2015*

- Start with sparse flow field from key point matching
- Calculate "edge aware distance" between two pixels p and q

$$D_G(\boldsymbol{p}, \boldsymbol{q}) = \inf_{\Gamma \in \mathcal{P}_{\boldsymbol{p},\boldsymbol{q}}} \int_\Gamma C(\boldsymbol{p}_s) d\boldsymbol{p}_s$$

- Then perform weighted average interpolation for flow at pixel p

$$\boldsymbol{F}_{NW}(\boldsymbol{p}) = \frac{\sum_{(\boldsymbol{p}_m, \boldsymbol{p}'_m) \in \mathcal{M}} k_D(\boldsymbol{p}_m, \boldsymbol{p}) \boldsymbol{p}'_m}{\sum_{(\boldsymbol{p}_m, \boldsymbol{p}'_m) \in \mathcal{M}} k_D(\boldsymbol{p}_m, \boldsymbol{p})}$$

- Finally smooth over flow image using one level variational method

$$E(w) = E_{\text{data}} + \alpha E_{\text{smoothness}}$$

EpicFlow - KITTI

Table : KITTI Results

Inputs

Matching

Edges

Result

Source: *Revaud et al, CVPR 2015*

Difference between Scene Flow and Optical FLow

- Optical flow reasons in 2D
  - Output is only flow field showing 2D pixel correspondences
- Scene flow reasons in 3D
  - Recognizes that changes in the 2D image are due to object motion and/or change in camera position in 3D
  - Optical flow = projection of 3D scene flow onto image
  - Output can include:
    - 3D structure (depth)
    - 3D Camera motion
    - 3D Object motion

Why bother with scene flow?

- Optical flow reasons about things like matching points, edges, interpolation, regional smoothing, etc in 2D
    - Can only incorporate intuition about 2D
- But we know much about the world in 3D! For example, we know that in general...
    - Many (most) objects are rigid
    - Many (most) surfaces are planar
    - Objects have continuous motion - no teleportation (in classical physics)
- Plus some domain specific knowledge (eg for driving):
    - Most of a scene is likely static (road, trees, buildings)
    - Most object motion is confined to road plane (for now)
    - Camera motion is confined to road plane (if car is driven well)
- How do we incorporate these beliefs?
    - Reason in 3D!

Why bother with scene flow?

Source: *KITTI*

Why is Scene Flow Hard?

- Inherent ambiguity:
  - Need to infer 3D information from 2D images
  - 2D observed flow can be due to any combination of object motion, camera motion, depth
- Similar problems as optical flow
  - Sparseness of matches
  - Errors in matching
  - Occlusions, displacement, view point changes, etc

3D to 2D Projection Review

- Given a 3D point $\mathbf{P} = [X, Y, Z, 1]^\intercal$ in homogeneous coordinates
- Projection into image is $\mathbf{p} = [x, y, 1]^\intercal$

$$\mathbf{p} = [K|0][R|\vec{t}]\mathbf{P} \tag{1}$$

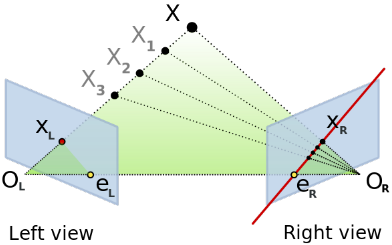
- For canonical camera at world center: $K = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & f \end{bmatrix}$, $R = I$, $\vec{t} = 0$
  - and thus $x = f\frac{X}{Z}$, $y = f\frac{Y}{Z}$

Epipolar Geometry Review

- Given two images I and I' of a **static** scene, p' corresponding to p must lie on epipolar line:

$$\mathbf{p}^{\mathsf{T}} F \mathbf{p'} = 0$$

- $F \subset \mathbb{R}^{3x3}$ = fundamental matrix, rank = 2



Source: *IMAGINE @ http://imagine.enpc.fr/*

Robust Monocular Epipolar Flow Estimation

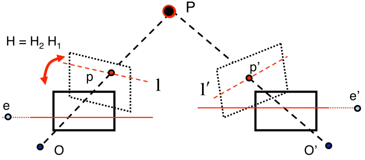
- Robust Monocular Epipolar Flow Estimation
  - Yamaguchi, McAllester, Urtasun, CVPR2013
- Scene flow estimation from monocular (not stereo) image pairs at different times
- Assumes static scene
- Based on
  - Epipolar flow
  - Planar superpixel assumption
  - Smoothing

Epipolar Flow

- Optical flow assumed to consist of two components
- $\mathbf{p}$ and $\mathbf{p}'$ are corresponding points in I and I'
- Flow $\mathbf{u} = \mathbf{p}' - \mathbf{p}$
- $\mathbf{u} = \mathbf{u_w}(\mathbf{p}) + \mathbf{u_v}(\mathbf{p})$ has two components:
  - $\mathbf{u_w}(\mathbf{p})$ due to camera rotation
  - $\mathbf{u_v}(\mathbf{p})$ due to camera translation
- Idea:
  - Undo camera rotation first (by estimating F)
  - Treat problem as epipolar flow

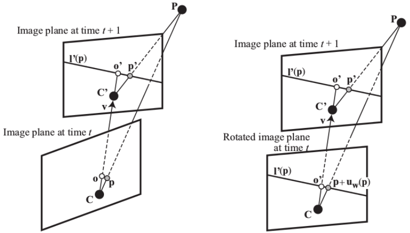
Epipolar Flow - Rectification

- Stereo rectification



Source: *S. Savarese*

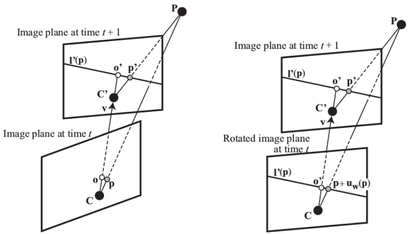
- Epipolar rectification



Source: *Yamaguchi et al*

Epipolar Flow - Rectification

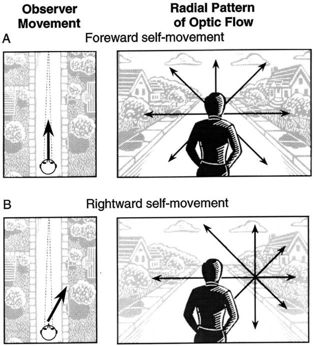
- Epipolar expansion (or contraction)
  - Only valid when camera motion is purely translational!
  - Flow is confined to epipolar lines coming out of epipole
  - Epipole is found by right nullspace of F
    - $Fo' = 0$
  - Rectified image pair have epipole at SAME location
  - Knowing F and o' we can find epipolar line in I' on which $\mathbf{p}'$ must lie



Source: *Yamaguchi et al*

Epipolar Flow - Searching for Matches

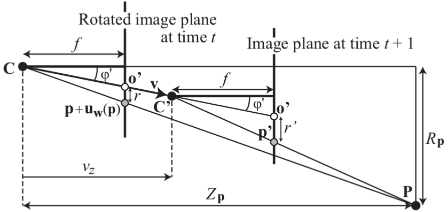
- Search along epipolar lines to find matching point pairs
- Generate dense matches (flow field)
  - Minimize matching cost using Semi Global Smoothing
  - (for more info: http://www.ifp.uni-stuttgart.de/publications/phowo11/180Hirschmueller.pdf)



Source: *neurology.org*

Epipolar Flow - Segmentation and Smoothing

- Given motion is only translation + static scene
  - constant v_z for all 3D points relative to camera

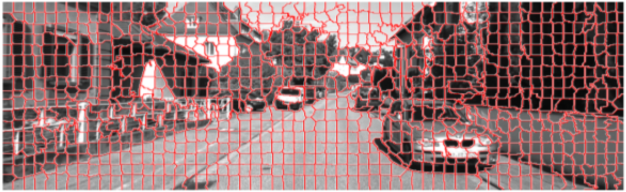


- Define a ratio $\omega_\mathbf{p} = \frac{v_\mathbf{p}}{Z_\mathbf{p}}$ called the vz-index for each point $\mathbf{p}$

Source: *Yamaguchi et al*

Epipolar Flow - Segmentation and Smoothing

- Assume each piecewise planar world - superpixels represent 3D planes
- Jointly segment image into slanted plane superpixels using
  - Pixel appearance + location
  - Flow information (use the vz-index for each **p**)
  - Regularizers



- Planar assumption constrains vz-ratios for all $\mathbf{p} = (u, v)$ belonging to superpixel i centered at c_i to disparity plane:
  - $\omega_i = \alpha_i(u - c_{iv}) + \beta_i(v - c_{iv}) + \gamma_i$
- Goal - fit one such disparity plane described by α_i, β_i, γ_i to each superpixel

Source: *Yamaguchi et al*

Question

- What about scenes with moving objects?
- Points in general no longer obey the same epipolar constraint
- How can we estimate both world geometry AND object motion?

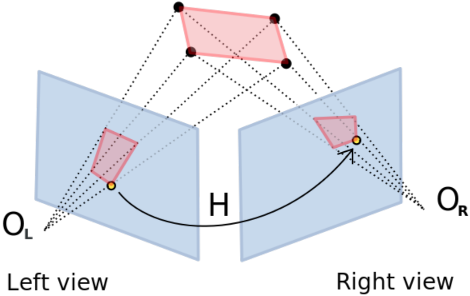
3D Piecewise Rigid Scene Model

- 3D Scene Flow Estimation with a Piecewise Rigid Scene Model
    - Vogel et al, IJCV 2015
- Scene flow estimation from stereo image pairs + time series of images
- Handles dynamic environment
- Based on
    - Homographies
    - Piecewise planar assumption of 3D world
    - Single reference view energy minimization
    - Multi-frame motion continuity

3D Piecewise Rigid Scene Model - Homography Review

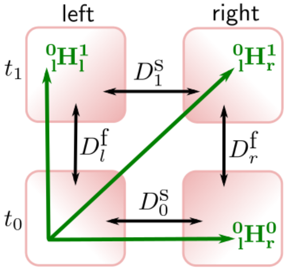
- Similar idea to epipolar geometry and fundamental matrix, but different
- Fundamental Matrix:
  - $F \subset \mathbb{R}^{3x3}$, rank 2 maps point $\mathbf{p}$ in I to **line** $l' = F\mathbf{p}$ in I'
  - Valid for general 3D points in world
- Homography Matrix
  - $H \subset \mathbb{R}^{3x3}$, rank 3 maps point $\mathbf{p}$ in I to **point** $\mathbf{p}' = H\mathbf{p}$ in I'
  - Valid for 3D points in world belonging to same plane $n^\intercal \mathbf{P} = 0$

- "Homography" means "similar drawing"



Source: *IMAGINE @ http://imagine.enpc.fr/*

- Let $\pi = \pi(R, t, \vec{n})$ be a moving plane in 3D with 9 total degrees of freedom
  - 3 degrees in each of R, t, $\vec{n}$
- Two observing cameras l and r taking two pictures each at $t = 0$ and $t = 1$
  - Camera l at $t = 0$ considered canonical, aka $p_l^0 = [K|0]\mathbf{P}^0$
  - Camera r at $t = 0$ has projection matrix M, or $p_r^0 = [M|m]\mathbf{P}^0$
- Thus view of plane can be transformed between view points and time:

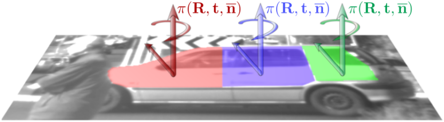


$${}^0_l\mathbf{H}^0_r(\pi) = (\mathbf{M} - \mathbf{m}\bar{\mathbf{n}}^\top)\mathbf{K}^{-1}$$
$${}^0_l\mathbf{H}^1_l(\pi) = \mathbf{K}(\mathbf{R} - \mathbf{t}\bar{\mathbf{n}}^\top)\mathbf{K}^{-1}$$
$${}^0_l\mathbf{H}^1_r(\pi) = (\mathbf{M}\mathbf{R} - (\mathbf{M}\mathbf{t} + \mathbf{m})\bar{\mathbf{n}}^\top)\mathbf{K}^{-1}$$

Source: *Vogel et al*

3D Piecewise Rigid Scene Model - Technique

- Method assumes scene is piecewise planar



- Fits parameters to each plane by energy minimization involving:
  - Single reference image (2 cameras, 2 times), or more generally
  - Entire image sequence terms (2 cameras, consistent motion through time)

Source: *Vogel et al*

3D Piecewise Rigid Scene Model - Technique

Single reference image (2 cameras, 2 times)

- Let $\mathcal{P}$ be the assignment of pixels to segments
- Let $\mathcal{S}$ be the assignment of segments to 3D moving plane

$$E(\mathcal{P},\mathcal{S}) = E_{\mathbf{D}}(\mathcal{P},\mathcal{S}) + \lambda E_{\mathbf{R}}(\mathcal{P},\mathcal{S}) + \mu E_{\mathbf{S}}(\mathcal{P}) + E_{\mathbf{V}}(\mathcal{P},\mathcal{S})$$

- To initialize inference, use output of other optical / stereo / scene flow algorithms
- Energy here essentially further refines flow by adding additional priors

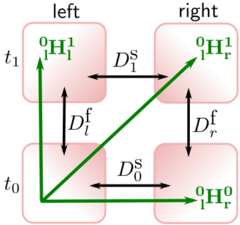
3D Piecewise Rigid Scene Model - Technique

Single reference image (2 cameras, 2 times)

- Let $\mathcal{P}$ be the assignment of pixels to segments
- Let $\mathcal{S}$ be the assignment of segments to 3D moving plane

$$E(\mathcal{P},\mathcal{S}) = E_{\mathbf{D}}(\mathcal{P},\mathcal{S}) + \lambda E_{\mathbf{R}}(\mathcal{P},\mathcal{S}) + \mu E_{\mathbf{S}}(\mathcal{P}) + E_{\mathbf{V}}(\mathcal{P},\mathcal{S})$$

$E_{\mathbf{D}}(\mathcal{P},\mathcal{S})$ encourages view consistency of planes between cameras and time



Source: *Vogel et al*

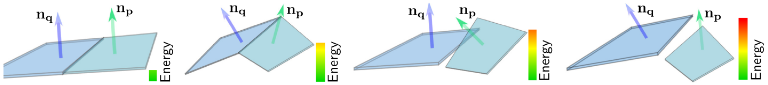
3D Piecewise Rigid Scene Model - Technique

Single reference image (2 cameras, 2 times)

- Let $\mathcal{P}$ be the assignment of pixels to segments
- Let $\mathcal{S}$ be the assignment of segments to 3D moving plane

$$E(\mathcal{P},\mathcal{S}) = E_{\mathbf{D}}(\mathcal{P},\mathcal{S}) + \lambda E_{\mathbf{R}}(\mathcal{P},\mathcal{S}) + \mu E_{\mathbf{S}}(\mathcal{P}) + E_{\mathbf{V}}(\mathcal{P},\mathcal{S})$$

$E_{\mathbf{R}}(\mathcal{P},\mathcal{S})$ builds in a prior favoring simpler scene geometry and motion



Source: *Vogel et al*

3D Piecewise Rigid Scene Model - Technique

Single reference image (2 cameras, 2 times)

- Let $\mathcal{P}$ be the assignment of pixels to segments
- Let $\mathcal{S}$ be the assignment of segments to 3D moving plane

$$E(\mathcal{P},\mathcal{S}) = E_{\mathbf{D}}(\mathcal{P},\mathcal{S}) + \lambda E_{\mathbf{R}}(\mathcal{P},\mathcal{S}) + \mu E_{\mathbf{S}}(\mathcal{P}) + E_{\mathbf{V}}(\mathcal{P},\mathcal{S})$$

$E_{\mathbf{S}}(\mathcal{P})$ sets maximum superpixel size so that scene model doesn't become overly simple

3D Piecewise Rigid Scene Model - Technique

Single reference image (2 cameras, 2 times)

- Let $\mathcal{P}$ be the assignment of pixels to segments
- Let $\mathcal{S}$ be the assignment of segments to 3D moving plane

$$E(\mathcal{P},\mathcal{S}) = E_{\mathbf{D}}(\mathcal{P},\mathcal{S}) + \lambda E_{\mathbf{R}}(\mathcal{P},\mathcal{S}) + \mu E_{\mathbf{S}}(\mathcal{P}) + E_{\mathbf{V}}(\mathcal{P},\mathcal{S})$$

$E_{\mathbf{V}}(\mathcal{S})$ - based on estimation for changes in visibility of areas of image, penalize inconsistent motion proposals

Extension to View-Consistent Model (2 cameras, more than 2 times)

- Prior: objects don't suddenly teleport within sequence of frames
- Prior: objects have non-zero mass (no instantaneous acceleration, smooth velocity)
- Let $\mathcal{P}$ be the assignment of pixels to segments
- Let $\mathcal{S}$ be the assignment of segments to 3D moving plane

$$E(\mathcal{P},\mathcal{S}) = E_{\mathbf{D}}^{VC}(\mathcal{P},\mathcal{S}) + \lambda E_{\mathbf{R}}^{VC}(\mathcal{P},\mathcal{S}) + \mu E_{\mathbf{S}}^{VC}(\mathcal{P})$$

3D Piecewise Rigid Scene Model - Technique

Extension to View-Consistent Model (2 cameras, more than 2 times)

- Let $\mathcal{P}$ be the assignment of pixels to segments
- Let $\mathcal{S}$ be the assignment of segments to 3D moving plane

$$E(\mathcal{P},\mathcal{S}) = E_{\mathbf{D}}^{VC}(\mathcal{P},\mathcal{S}) + \lambda E_{\mathbf{R}}^{VC}(\mathcal{P},\mathcal{S}) + \mu E_{\mathbf{S}}^{VC}(\mathcal{P})$$

$E_{\mathbf{D}}^{VC}(\mathcal{P},\mathcal{S})$ now encourages:

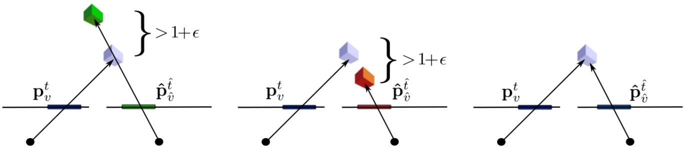
- View consistency between cameras + times (same as before)
- Plausibility of occlusions
- Fewer cases of moving out of bounds
- Fewer moving plane violations

Extension to View-Consistent Model (2 cameras, more than 2 times)

- Let $\mathcal{P}$ be the assignment of pixels to segments
- Let $\mathcal{S}$ be the assignment of segments to 3D moving plane

$$E(\mathcal{P},\mathcal{S}) = E_{\mathbf{D}}^{VC}(\mathcal{P},\mathcal{S}) + \lambda E_{\mathbf{R}}^{VC}(\mathcal{P},\mathcal{S}) + \mu E_{\mathbf{S}}^{VC}(\mathcal{P})$$

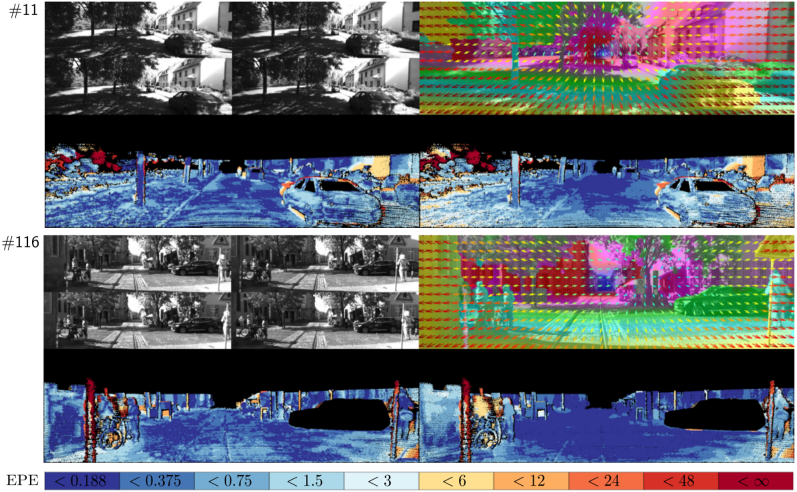
$E_{\mathbf{D}}^{VC}(\mathcal{P},\mathcal{S})$



(a) Implausible case **(b)** Occlusion **(c)** Normal case

Source: *Vogel et al*

3D Piecewise Rigid Scene Model - Sample Result



Source: *Vogel et al*

Wrap-up

- Thanks!
- Questions?
- Suggestions for future directions?