GPU Programming in Computer Vision

Final Projects

Thomas Möllenhoff, Robert Maier, Caner Hazirbas, Lingni Ma

Winter Semester 2015/2016

Project Phase (March 21 - April 13)

- Form groups of 3 people

- Implement a computer vision algorithm in CUDA
  - Select your 5 favorite topics (ordered by preference)
  - We will assign the projects to the groups

- Regular meetings with your supervisor
- Send source code to your supervisor until April 16

- Cheating: all involved groups will get the grade 5.0

Presentations (April 14/15)

- 15 minutes per group
- Prepare slides
  - Explain the task
  - Explain how you proceeded to solve the task
  - Show your results
- Live demo
- Q&A session

Final Project Proposals

Implement your own project idea?

1) Preconditioned Primal-Dual Method (Frank R. Schmidt)
2) Nonlinear Shape Registration (Csaba Domokos)
3) Nonlinear Spectral Image Fusion (Michael Möller)
4) Shading-based Refinement of Depth Images (Christian Kerl)
5) Dense Visual Odometry (Georg Kuschk)
6) Dense SLAM Framework Improvements (Georg Kuschk)
7) RGB-D Keyframe Fusion (Robert Maier)

GPU Programming in Computer Vision (IN2106)

Frank R. Schmidt

Winter Semester 2015/2016

Linear Programming

Many problems in Computer Vision can be cast as a **Linear Program (LP)**:

$$\left\{\begin{matrix}\min \\ \max\end{matrix}\right\}_{x \in \mathbb{R}^n} \langle c, x \rangle$$

$$\text{subject to} \langle a_i, x \rangle \left\{\begin{matrix}\leqslant \\ = \\ \geqslant\end{matrix}\right\} b_i \qquad \text{for all } i = 1, \ldots, m$$

Many problems in Computer Vision can be cast as a **Linear Program (LP)**:

$$\begin{Bmatrix} \min \\ \max \end{Bmatrix}_{x \in \mathbb{R}^n} \langle c, x \rangle$$

$$\text{subject to} \langle a_i, x \rangle \begin{Bmatrix} \leqslant \\ = \\ \geqslant \end{Bmatrix} b_i \qquad \text{for all } i = 1, \ldots, m$$

Any LP can be transformed into the form

$$\min_{x \in \mathbb{R}^n} \langle c, x \rangle$$
$$\text{subject to } Ax = b$$
$$x \geqslant 0$$

Preconditioned Primal-Dual Method

The goal of the assignment is the re-implementation of the preconditioned primal-dual method:

$$
\begin{aligned}
x^{(k+1)} =& \pi_{\geqslant 0}\left(x^{(k)} - T(A^\top y^{(k)} + c)\right) \\
y^{(k+1)} =& y^{(k)} + S\left(A\left(2x^{(k+1)} - x^{(k)}\right) - b\right)
\end{aligned}
$$

Preconditioned Primal-Dual Method

The goal of the assignment is the re-implementation of the preconditioned primal-dual method:

$$x^{(k+1)} = \pi_{\geqslant 0}\left(x^{(k)} - T(A^\top y^{(k)} + c)\right)$$
$$y^{(k+1)} = y^{(k)} + S\left(A\left(2x^{(k+1)} - x^{(k)}\right) - b\right)$$

T. Pock and A. Chambolle, *"Diagonal preconditioning for first order primal-dual algorithms in convex optimization"*, 2011, IEEE ICCV, 1762–1769.

Preconditioned Primal-Dual Method

The goal of the assignment is the re-implementation of the preconditioned primal-dual method:

$$
\begin{aligned}
x^{(k+1)} =& \pi_{\geqslant 0}\left(x^{(k)} - T(A^\top y^{(k)} + c)\right) \\
y^{(k+1)} =& y^{(k)} + S\left(A\left(2x^{(k+1)} - x^{(k)}\right) - b\right)
\end{aligned}
$$

T. Pock and A. Chambolle, *"Diagonal preconditioning for first order primal-dual algorithms in convex optimization"*, 2011, IEEE ICCV, 1762–1769.

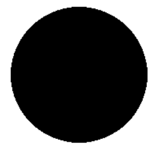
Applications may include

- general LP
- shape matching
- image segmentation

Nonlinear Shape Registration

Problem definition

Registration: we aim to find the *geometric deformation* between two shapes.

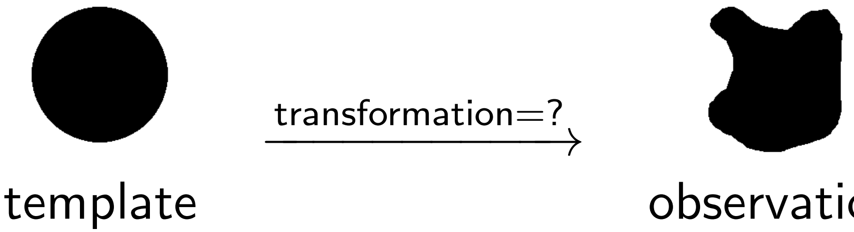


template

observation

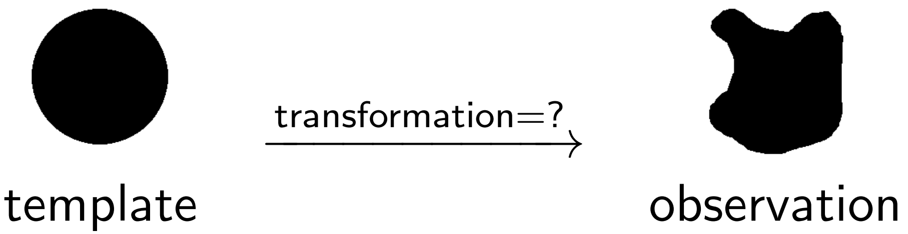
Problem definition

Registration: we aim to find the *geometric deformation* between two shapes.



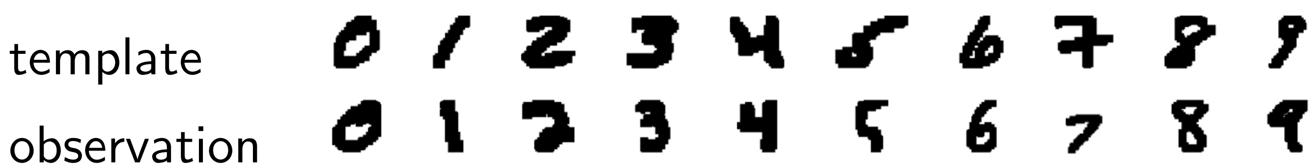
Problem definition

Registration: we aim to find the *geometric deformation* between two shapes.



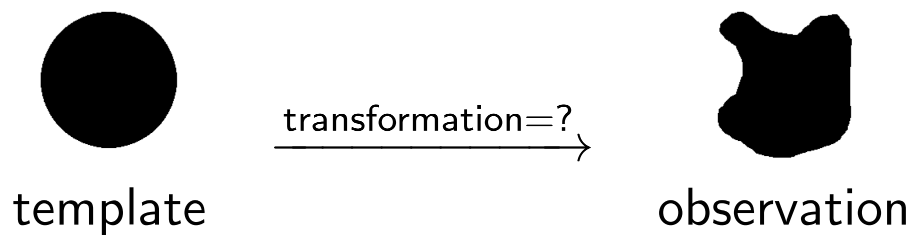
There are *various applications* of this problem:

Registration of handwritten characters



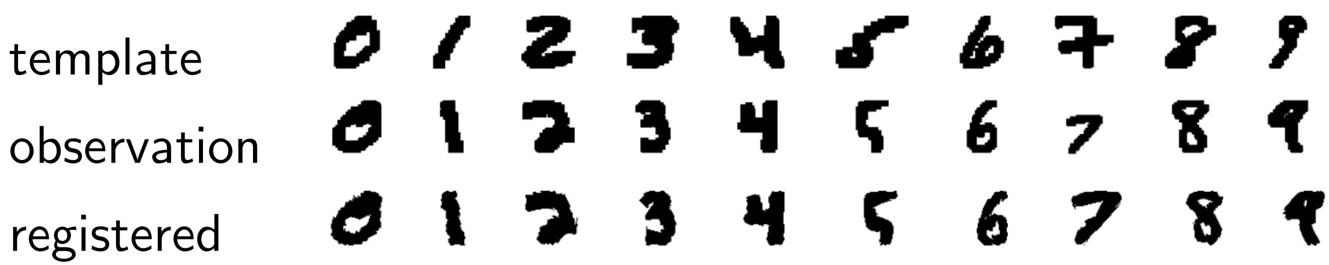
Problem definition

Registration: we aim to find the *geometric deformation* between two shapes.

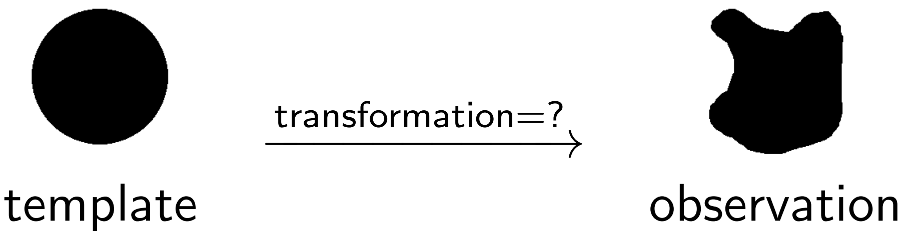


There are *various applications* of this problem:

Registration of handwritten characters

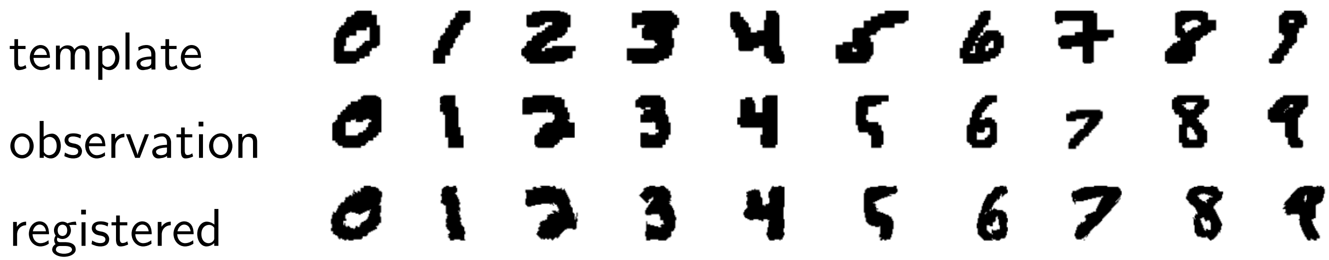


Registration: we aim to find the *geometric deformation* between two shapes.

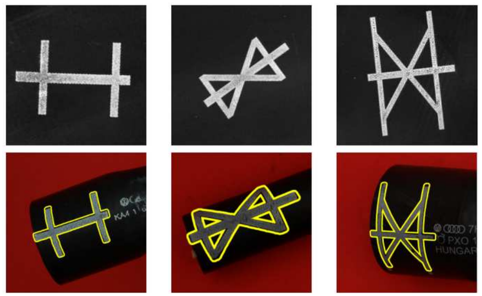


There are *various applications* of this problem:

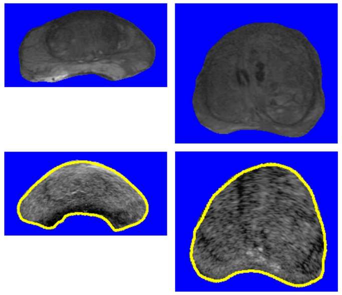
Registration of handwritten characters



Industrial inspection



Medical image registration



1. Calculate **image moments**, i.e. sum of the powers of pixel coordinates over the foreground region $\mathcal{F}$, that is

$$m_{ij} = \int\int x^i y^j \mathbb{1}_{\mathcal{F}}(x,y) \mathrm{d}x \mathrm{d}y \approx \sum_{(x,y)\in\mathcal{F}} x^i y^j \qquad \text{where } i,j \in \mathbb{N}\,.$$

Sub-tasks of the problem

1. Calculate **image moments**, i.e. sum of the powers of pixel coordinates over the foreground region $\mathcal{F}$, that is

$$m_{ij} = \int\int x^i y^j \mathbb{1}_{\mathcal{F}}(x,y)\mathrm{d}x\mathrm{d}y \approx \sum_{(x,y)\in\mathcal{F}} x^i y^j \qquad \text{where } i,j \in \mathbb{N}\,.$$

2. Generate the image of a (nonlinear) **geometric transformation** applied to a shape.

Sub-tasks of the problem

1. Calculate **image moments**, i.e. sum of the powers of pixel coordinates over the foreground region $\mathcal{F}$, that is

$$m_{ij} = \int\int x^i y^j \mathbb{1}_{\mathcal{F}}(x,y)\mathrm{d}x\mathrm{d}y \approx \sum_{(x,y)\in\mathcal{F}} x^i y^j \qquad \text{where } i,j \in \mathbb{N}\ .$$

2. Generate the image of a (nonlinear) **geometric transformation** applied to a shape.
3. Solve a *system of nonlinear equations* via **Levenberg-Marquardt algorithm**, which is a combination of *Gauss–Newton method* and *gradient descent*.
   - Standard algorithm for *non-linear least squares* problems:

$$\underset{p}{\operatorname{argmin}}\, F(p) = \underset{p}{\operatorname{argmin}} \sum_{i=1}^{m} \left(c_i - f(d_i;p)\right)^2 .$$

   - Note that there already exist some implementation in `CUDA`.

Sub-tasks of the problem

1. Calculate **image moments**, i.e. sum of the powers of pixel coordinates over the foreground region $\mathcal{F}$, that is

$$m_{ij} = \int\int x^i y^j \mathbb{1}_{\mathcal{F}}(x,y)\mathrm{d}x\mathrm{d}y \approx \sum_{(x,y)\in\mathcal{F}} x^i y^j \qquad \text{where } i,j \in \mathbb{N}\,.$$

2. Generate the image of a (nonlinear) **geometric transformation** applied to a shape.
3. Solve a *system of nonlinear equations* via **Levenberg-Marquardt algorithm**, which is a combination of *Gauss–Newton method* and *gradient descent*.
   - Standard algorithm for *non-linear least squares* problems:

$$\operatorname*{argmin}_p F(p) = \operatorname*{argmin}_p \sum_{i=1}^{m} \left(c_i - f(d_i;p)\right)^2 .$$

   - Note that there already exist some implementation in `CUDA`.
4. *Optional*: implement an interface for `Python` or `Matlab`.

- The **paper** describing the method is available online:
  https://docs.google.com/file/d/0B6gqeZujyM56c1k4SGhaZzNjX1U/edit?usp=sharing

  C. Domokos, J. Nemeth and Z. Kato. **Nonlinear Shape Registration without Correspondences**. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 5, pp. 943-958, 2012.


Nonlinear Shape Registration without Correspondences

Supplementary materials

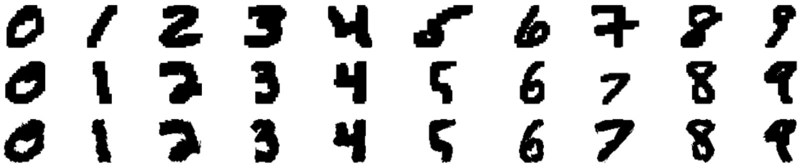
- The **paper** describing the method is available online:
  `https://docs.google.com/file/d/0B6gqeZujyM56c1k4SGhaZzNjX1U/edit?usp=sharing`

  C. Domokos, J. Nemeth and Z. Kato. **Nonlinear Shape Registration without Correspondences**. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 5, pp. 943-958, 2012.


Nonlinear Shape Registration without Correspondences
Csaba Domokos, József Nemeth, and Zoltan Kato, Senior Member, IEEE

- The **reference implementation** in `Matlab` is also available.
  (It will be shared with project members.)

In summary, you may learn an interesting problem and its mathematical background.



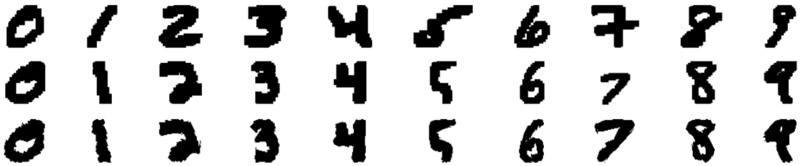
- The **paper** describing the method is available online:
  `https://docs.google.com/file/d/0B6gqeZujyM56c1k4SGhaZzNjX1U/edit?usp=sharing`

  C. Domokos, J. Nemeth and Z. Kato. **Nonlinear Shape Registration without Correspondences**. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 5, pp. 943-958, 2012.


Nonlinear Shape Registration without Correspondences

- The **reference implementation** in `Matlab` is also available. (It will be shared with project members.)

In summary, you may learn an interesting problem and its mathematical background.



Advisor: Dr. Csaba Domokos (`csaba.domokos@in.tum.de`)

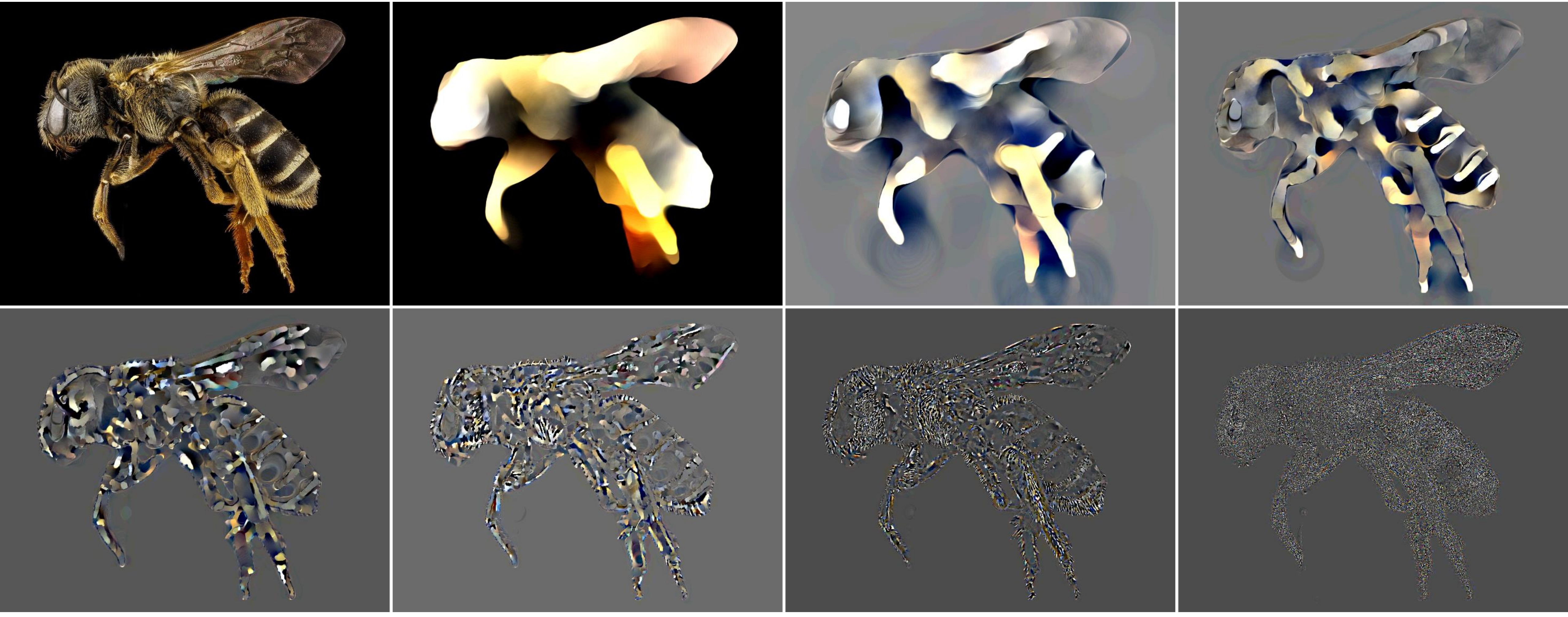
`https://vision.in.tum.de/members/domokos`

Please feel free to contact me!

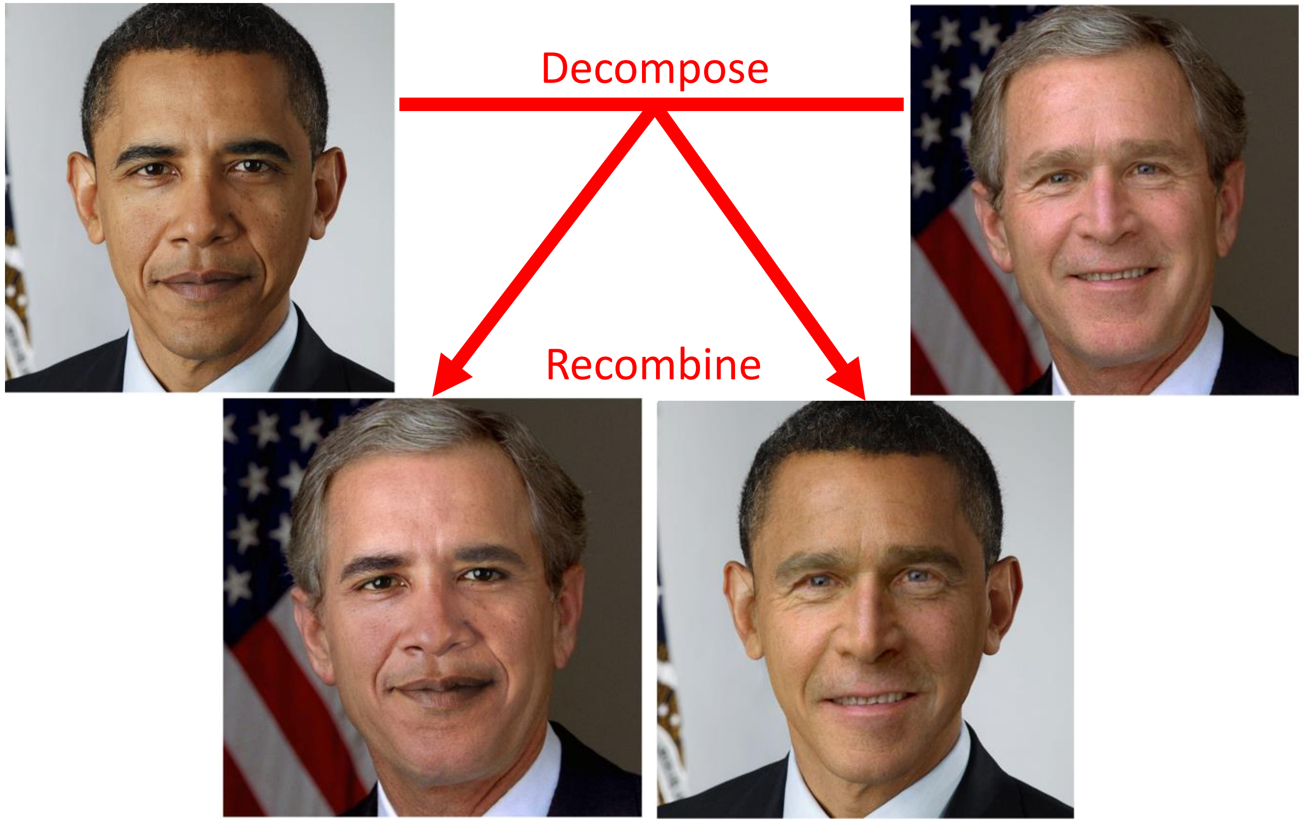
Nonlinear Spectral Image Fusion

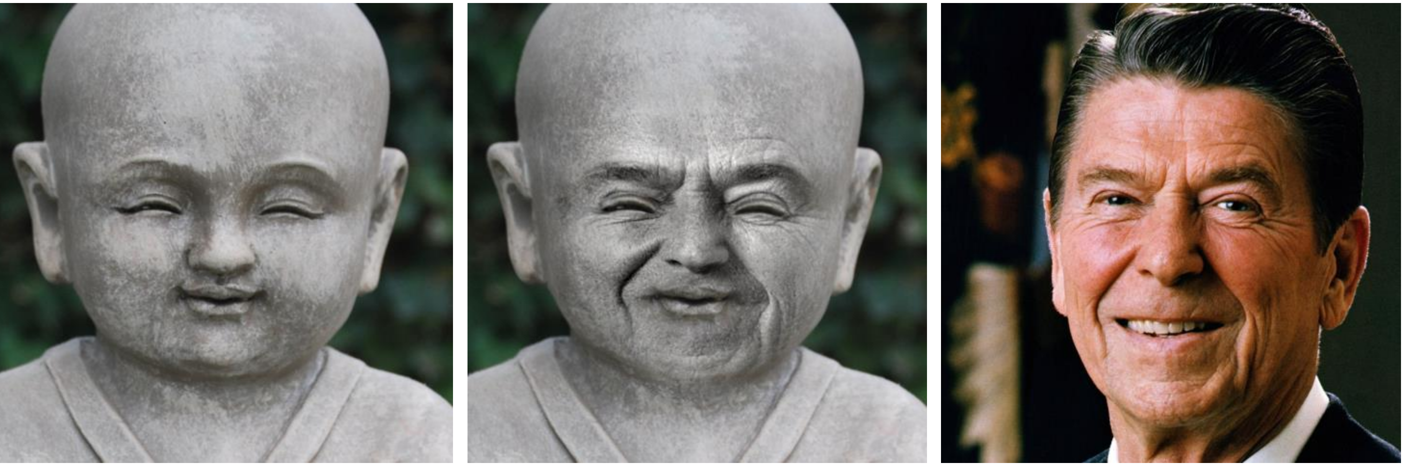
$$\partial_t p(t) = f - u(t) \quad \textbf{s.t. } p(t) \in \partial TV(u(t)),\ p(0) = 0$$

Main idea: By solving a sequence of total variation denoising problems, one gets a nice separation of image scales!



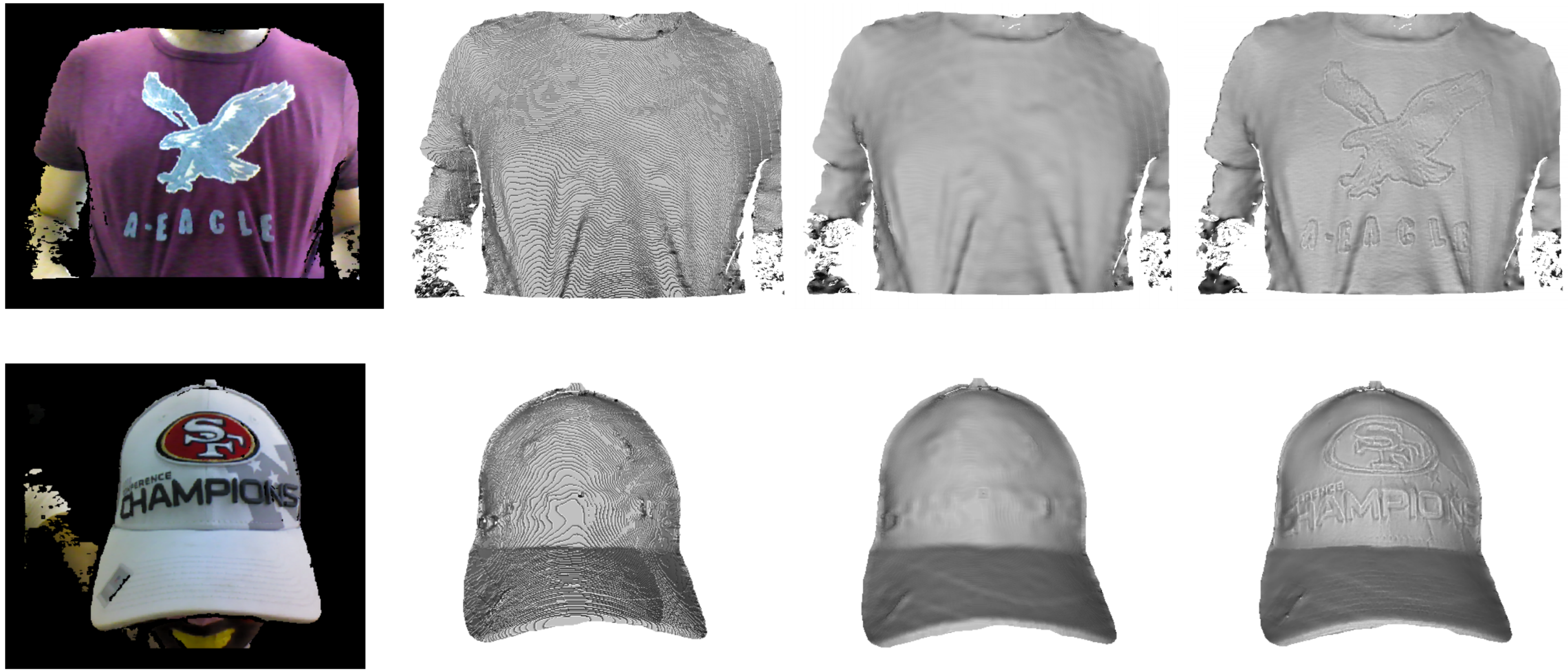
Recent application: Image Fusion





Shading-based Refinement of Depth Images

- Idea: Add details to depth image using appearance from RGB image

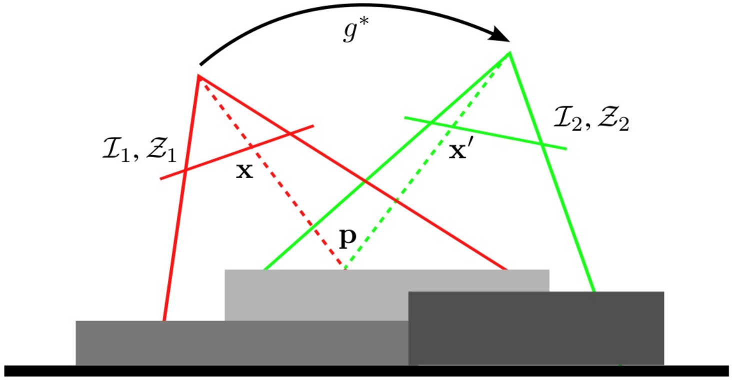


Shading-based Refinement of Depth Images

- Implementation
  - solve several large least squares problems using PCG
  - Matlab and CPU reference exist
- Papers
  - R. Or-El, G. Rosman, A. Wetzler, R. Kimmel and A. M. Bruckstein, "**RGBD-fusion: Real-time high precision depth recovery**", *CVPR 2015*
  - C. Wu, M. Zollhöfer, M. Nießner, M. Stamminger, S. Izadi, C. Theobalt, "**Real-time Shading-based Refinement for Consumer Depth Cameras**", SIGGRAPH 2014

Dense Visual Odometry

- Robust Odometry Estimation for RGB-D Cameras
  - Given: Two RGB-D frames

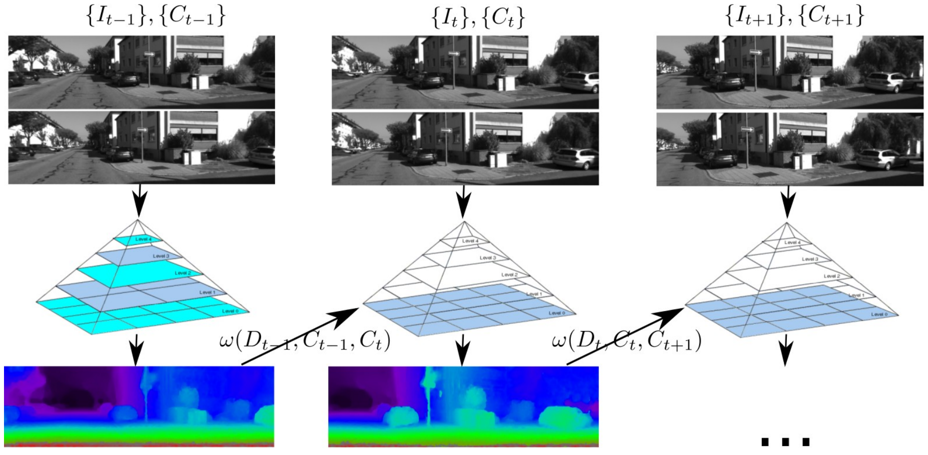


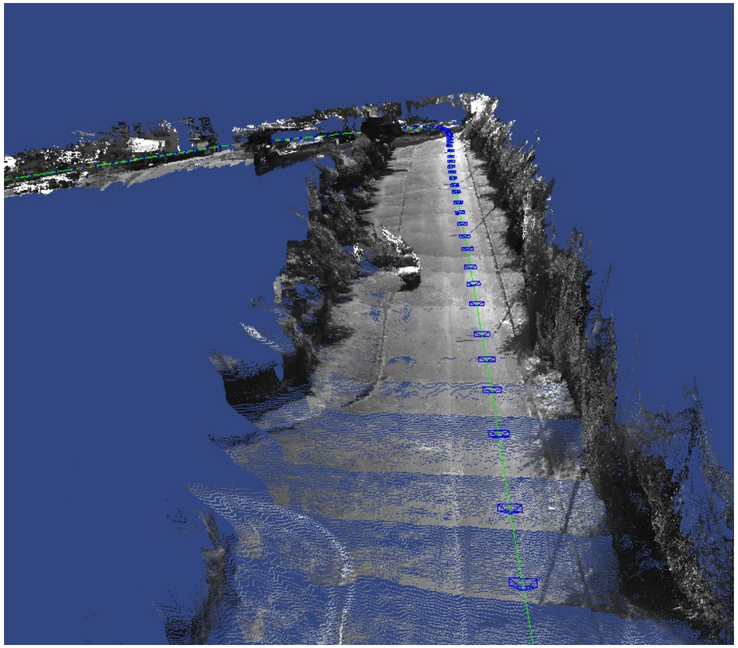
  - Goal: estimate camera motion g* by minimizing photometric and geometric error

- Given: Basic CPU and GPU implementation (320x240)
  → Goal:
- Clean and usable interface/code
- Refactor to high speed (ludicrous speed also possible)
- Implement additional functionality: Robust norms (Huber), student-t
- Reference: Robust Odometry Estimation for RGB-D Cameras [Kerl et al, ICRA 2013]

Improving existing CUDA code

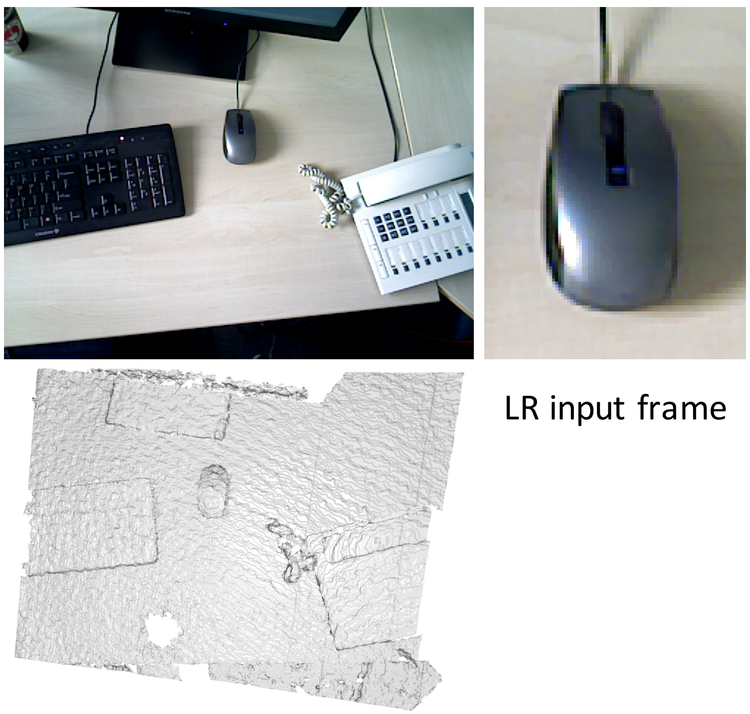
- For our dense SLAM framework a library of CUDA implementations exist
- Improve functions to ludicrous speed:
  - Interpolating images, warping of images, image resizing, primal-dual energy minimizations, …
  - Improve speed (CUDA textures, ...)
  - Improve accuracy (deterministic results)





RGB-D Keyframe Fusion

- Idea: fuse low-res. input RGB-D frames into high resolution RGB-D keyframes
  – Depth fusion (warp, upsample, fuse)
  – Color fusion (Deblur, warp, fuse)



LR input frame

Fused SR keyframe

Reference: Super-Resolution Keyframe Fusion for 3D Modeling with High-Quality Textures, Maier et al., 3DV 2015

Next steps

- Until March 20: send email to cuda-ws1516@cvpr.in.tum.de
  - Group Members
  - Your 5 favorite topics

- After project assignments: meet with your supervisor

- Any questions?