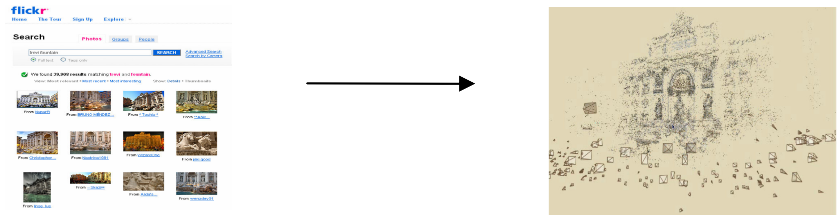


# Exploring Community Photo Collections

Professor and project’s advisor: Luiz Velho
Student: César Morais Palomo
cpalomo@inf.puc-rio.br

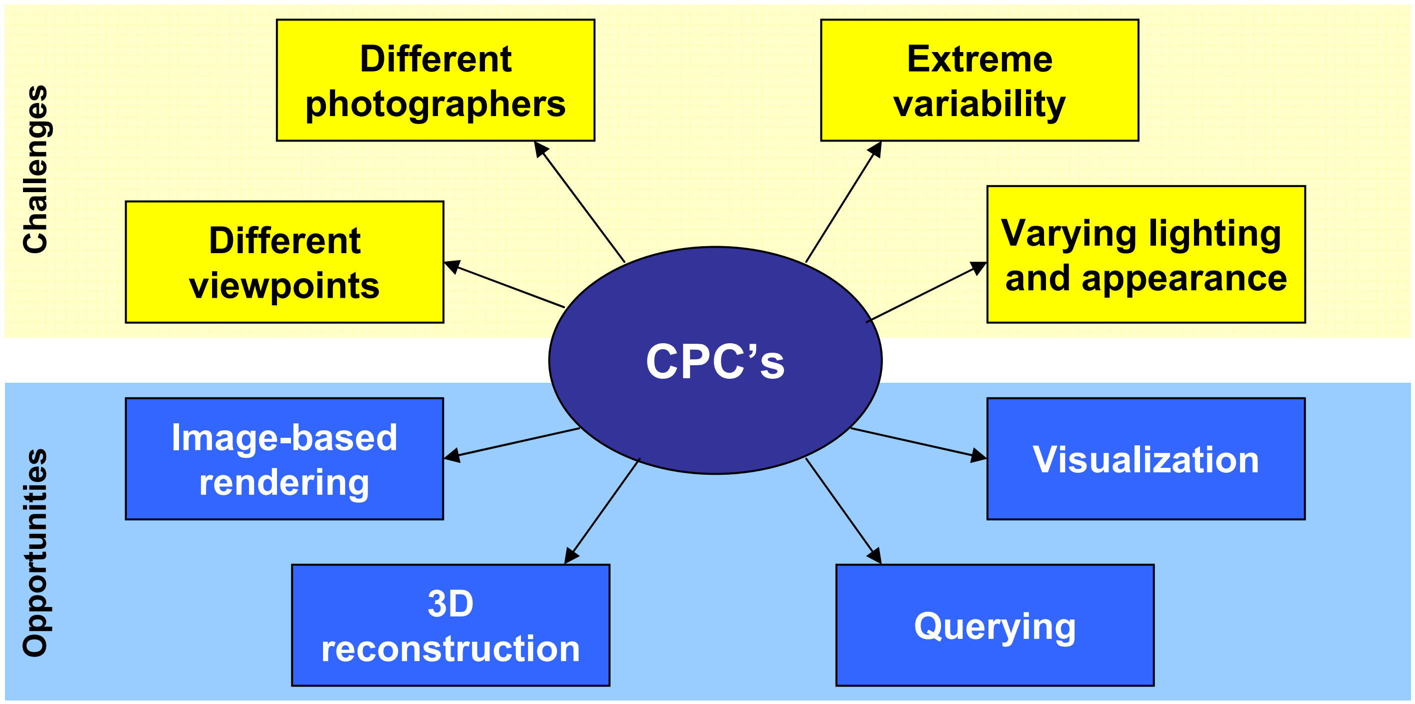
# Motivation

**Fact (= opportunity)**: more than 10 million members of the photo-sharing Web site Flickr snap pictures of their surroundings and then post those photos on the Internet

**Do the opposite**: download photos from Flickr and use them to recreate the original scenes

Google Images and Flickr: powerful new type of image dataset for computer vision and computer graphics research

# Characteristics of CPC's




**Objective**: find algorithms that operate robustly and successfully on such image sets to solve problems in computer vision and computer graphics

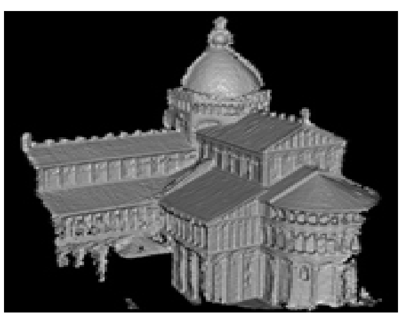
# Related work

**2006 Noah Snavely et al's *Photo Tourism***

- Developed for browsing large collections of photographs in 3D. It **automatically computes** each **photo's viewpoint** and a **sparse 3D model** of the scene. **Photo explorer** for moving about the 3D scene, through the photos.

**2008 Noah Snavely et al's *Finding Paths Through World Photos***

- Presents further **advances** in the **navigation control**. Exposes details of how to **discover** a set of paths for traversing **interesting regions and viewpoints of a scene**, and how to take advantage of them to **improve** the **user control** during image-based rendering.

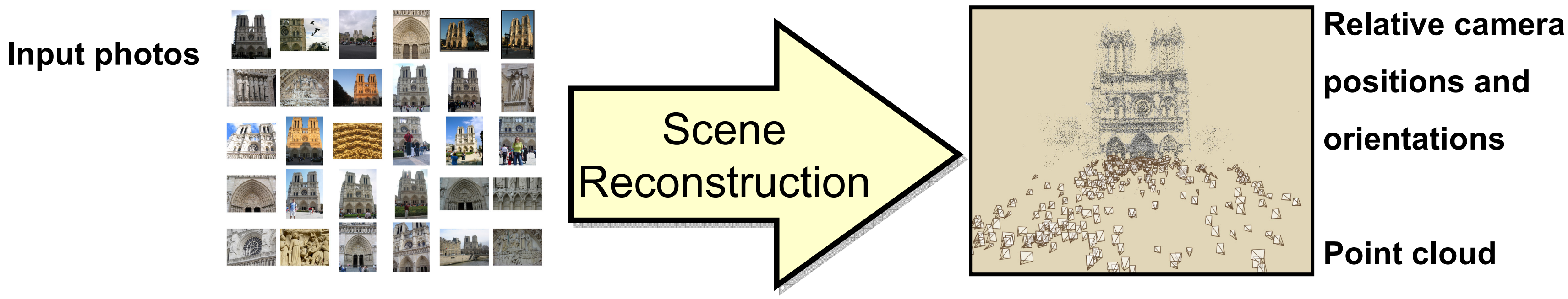


**2007 Goesele et al's *Multi-View Stereo for Community Photo Collections***

- Try to **reconstruct** the **3D geometry** of a scene from photo collections. Remarkable results.


cpalomo@inf.puc-rio.br

# Proposal




- Develop a full functioning **structure from motion** (***SfM***) framework to be used for **CPC's**, in a similar fashion to what has been done in *Photo Tourism* work.
- Follow the steps shown in the cited related work to validate the effectiveness of the ***SfM*** method.
- Make some minor modifications to the general proposed in order to try to improve performance.

# Method

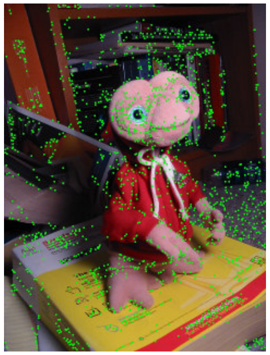
1. Features/keypoint detection in the input images using *SIFT*
2. Features matching for each pair of images
3. Fundamental matrix estimation using the eight-point algorithm, using *RANSAC*
4. Removal of matches that are outliers to the estimated fundamental matrix
5. Structure from motion step to estimate the parameters of each pair of cameras, in a bundle adjustment process


cpalomo@inf.puc-rio.br

# Method – Step 1

1. Features/keypoint detection in the input images using *SIFT*
2. Features matching for each pair of images
3. Fundamental matrix estimation using the eight-point algorithm, using *RANSAC*
4. Removal of matches that are outliers to the estimated fundamental matrix
5. Structure from motion step to estimate the parameters of each pair of cameras, in a bundle adjustment process

# Features detection



- Detect keypoints for each image using Lowe's *SIFT*

- To try to speed up this step, used Wu' *SiftGPU*
  - Implementation of SIFT for GPU
  - GPU shaders used in Gaussian pyramid construction, DoG keypoint detection and descriptor generation
  - Processes pixels and features paralelly in GPU and builds compact feature list by using GPU reduction: per-pixel processing changed to per-feature processing – reduces readback time

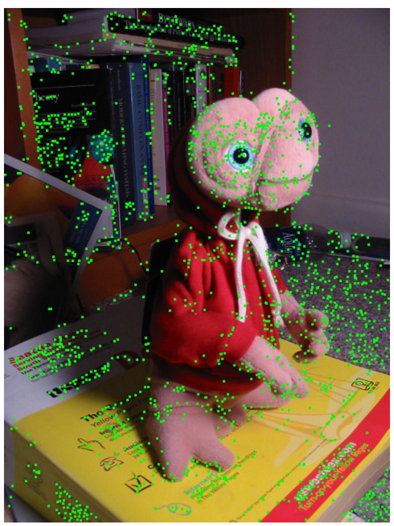
# Method – Step 2

1. Features/keypoint detection in the input images using *SIFT*
2. Features matching for each pair of images
3. Fundamental matrix estimation using the eight-point algorithm, using *RANSAC*
4. Removal of matches that are outliers to the estimated fundamental matrix
5. Structure from motion step to estimate the parameters of each pair of cameras, in a bundle adjustment process

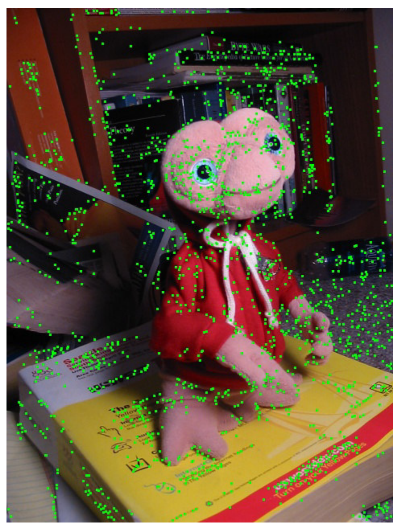

cpalomo@inf.puc-rio.br

# Features matching

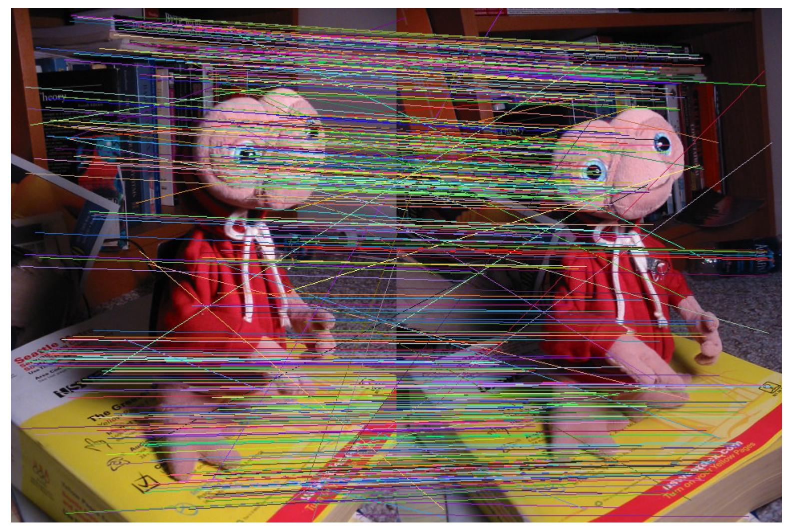
- SIFT descriptor: 128-dimension vector of integers
- Matching: try to identify corresponding features for each pair of images - relationship among photos



X



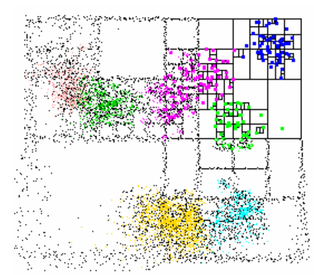
=




cpalomo@inf.puc-rio.br

# Robust features matching

- Naive approach for matching: brute-force computation of all distances for complete list of features for each pair of images
- Better try: use Mount's approximate nearest neighbors library to speed up search
  - Data structures and algorithms for exact and approximate nearest neighbor searching in arbitrarily high dimensions
  - Based on kd-trees and box-decomposition trees
  - Distances measured using any class of distance functions called Minkowski metrics




cpalomo@inf.puc-rio.br

# Method – Step 3

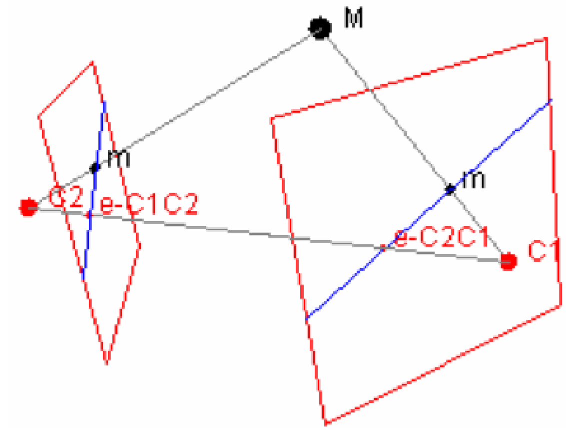
1. Features/keypoint detection in the input images using *SIFT*
2. Features matching for each pair of images
3. Fundamental matrix estimation using the eight-point algorithm, using *RANSAC*
4. Removal of matches that are outliers to the estimated fundamental matrix
5. Structure from motion step to estimate the parameters of each pair of cameras, in a bundle adjustment process

# Fundamental matrix

- Computer vision: **3x3** matrix of **rank 2** which relates corresponding points in stereo images: allows for detection of wrong correspondences

- Epipolar geometry: with corresponding points **m** and **m'** in a stereo image pair, **Fm** describes the epipolar line on which **m'** on the other image should lie:

$$m'^T Fm = 0 \qquad \begin{aligned} m &= [x \quad y \quad 1]^T \\ m' &= [x' \quad y' \quad 1]^T \end{aligned}$$




- Being of rank 2 and determined only up to scale, the F-matrix can be estimated given at least seven point correspondences

# F-matrix estimation: the 8-point algorithm

- Rewriting the equation: $$[xx' \ \ yx' \ \ x' \ \ xy' \ \ yy' \ \ y' \ \ x \ \ y \ \ 1]\,f = 0$$ $$f = [F_{11}\ F_{12}\ F_{13}\ F_{21}\ F_{22}\ F_{23}\ F_{31}\ F_{32}\ F_{33}]$$
- F-matrix: determined up to a scale factor
  - 8 equations are required to obtain a unique solution
- By stacking eight of these equations (8 point correspondences) in matrix A:

$$Af = 0$$

- Least-squares: perform SVD on A to get the eigenvector associated with the smallest singular value, i.e., the Null Space for A.

[U, S, V] = svd(A)

- Select the column of V that is associated with the least (or zero) singular value in S: the last column is our F_cand
- Enforce constraint that fundamental matrix has rank 2 by performing a SVD on F_cand and then reconstructing with the two largest singular values

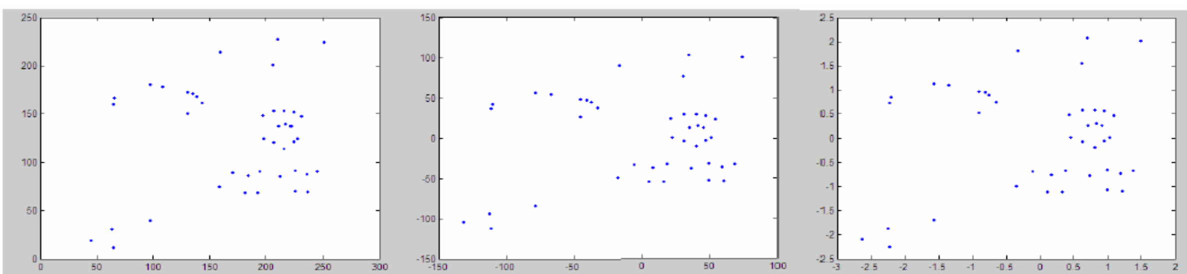
# F-matrix estimation: normalization

- F is often ill-conditioned: small variations in the data points (x,y coordinates) selected will completely mess up the calculation for F
  - i.e, magnitude of elements in **A** matters!

- How to solve or minimize this problem? Perform data normalization:
  - Translate mean location to origin
  - Scale so that average x and y distance to the origin is 1

$$\sum_{i=1}^{8} \frac{x_i}{8} = 0 \qquad \sum_{i=1}^{8} \frac{y_i}{8} = 0$$

$$\sum_{i=1}^{8} \frac{\sqrt{(x_i - \bar{x})^2 + (y_i - \bar{y})^2}}{8\sqrt{2}} = 1$$



$$\bar{x} = \sum_{i=1}^{8} \frac{x_i}{n} \qquad \bar{y} = \sum_{i=1}^{8} \frac{y_i}{n} \qquad d = \sum_{i=1}^{8} \frac{\sqrt{(x_i - \bar{x})^2 + (y_i - \bar{y})^2}}{8\sqrt{2}} \qquad T = \begin{pmatrix} 1/d & 0 & -\bar{x}/d \\ 0 & 1/d & -\bar{y}/d \\ 0 & 0 & 1 \end{pmatrix}$$

# F-matrix estimation: normalized 8-point algorithm

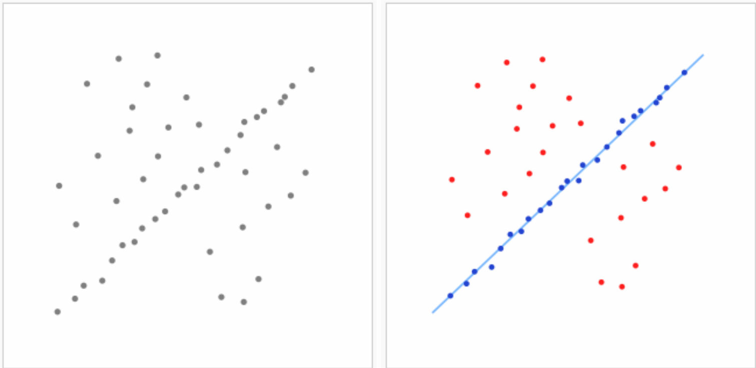
- Calculate normalization matrices for chosen points
- Use normalized points in the described 8-point algorithm
- Obtain the fundamental matrix for the original untransformed data by taking:

$$F = T_l^{\,T} \, F' T_r$$


cpalomo@inf.puc-rio.br

# RANSAC - **RAN**dom **SA**mple **C**onsensus

- **Iterative method** to estimate parameters of a mathematical model from a set of observed data which contains outliers
- A **non-deterministic** algorithm in the sense that it produces a reasonable result only with a certain probability, with this probability increasing as more iterations are allowed
- Very interesting for **robust estimation** of a model when data contains **outliers** and **noise**



# RANSAC general method

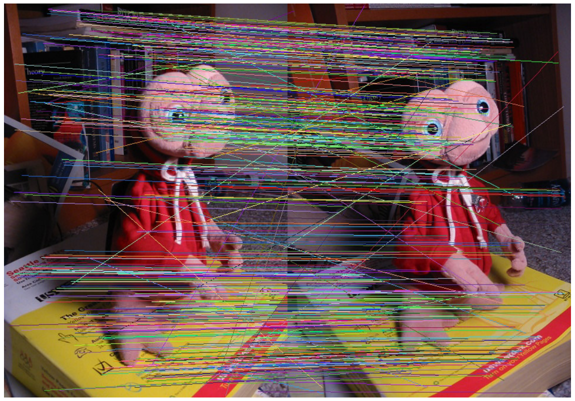
Repeat for a fixed number of iterations:

```
maybe_inliers := n randomly selected values from data
maybe_model := model parameters fitted to maybe_inliers
consensus_set := maybe_inliers

for every point in data not in maybe_inliers
    if point fits maybe_model with an error smaller than t
        add point to consensus_set

if number of elements in consensus_set is > d (good model, now test how good it is)
    better_model := model parameters fitted to all points in consensus_set
    this_error := a measure of how well better_model fits these points
    if this_error < best_error (best model found so far)
        best_model := better_model
        best_consensus_set := consensus_set
        best_error := this_error
```

# RANSAC for F-matrix estimation



Repeat for a fixed number of iterations:

```
maybe_inliers := 8 randomly selected matches
F_cand := normalized_8-point_algorithm using maybe_inliers
this_error = evaluate F_cand against all data
if this_error < best_error (best F found so far)
      F_best := F_cand
      best_error := this_error
```

After the fixed number of iterations, find inliers and outliers:

```
consensus_set := inliers for F_best, using distance to epipolar line as the err measure
outliers_set := all matches – consensus_set
```

How to evaluate F_cand:

```
error = 0
for each match (p1,p2) in all data
     error += findDistanceToEpipolarLine(p1, p2, F_cand) +
              findDistanceToEpipolarLine(p2, p1, transpose(F_cand))
```

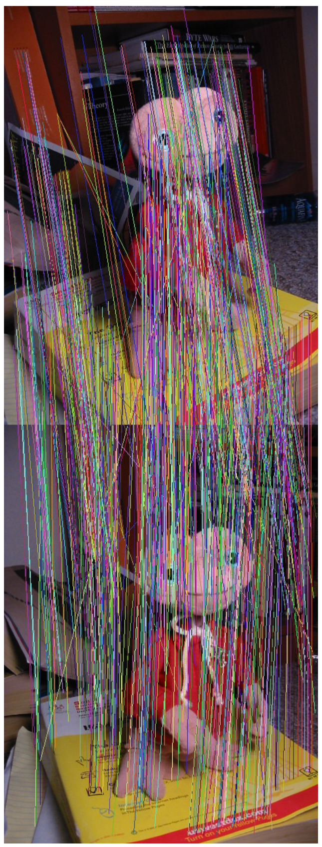
# Method – Step 4

1. Features/keypoint detection in the input images using *SIFT*
2. Features matching for each pair of images
3. Fundamental matrix estimation using the eight-point algorithm, using *RANSAC*
4. Removal of matches that are outliers to the estimated fundamental matrix
5. Structure from motion step to estimate the parameters of each pair of cameras, in a bundle adjustment process

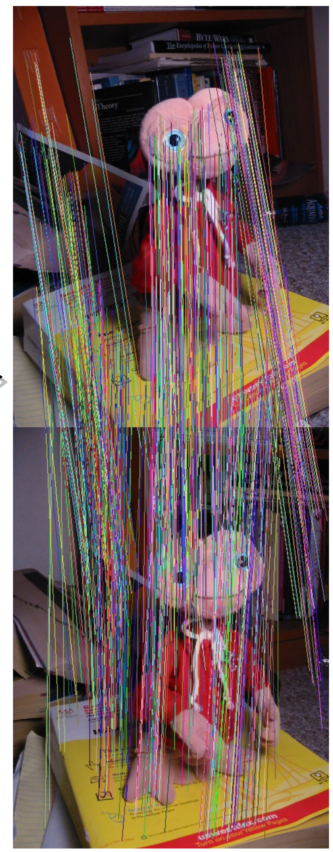
# Outliers removal by RANSAC

- RANSAC not only robustly fits a good F-matrix, but also remove outliers from consideration

All matches

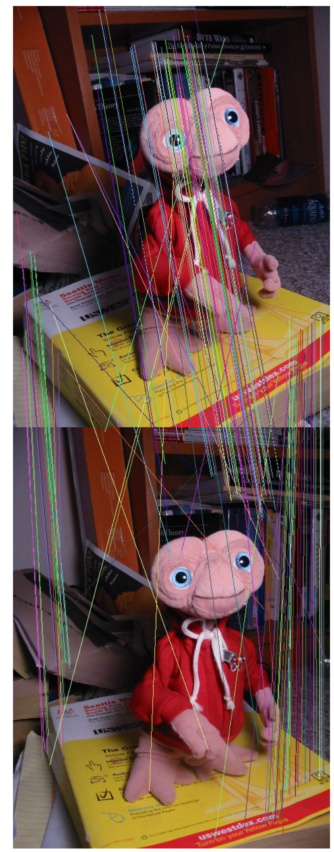


Inliers



+

Outliers

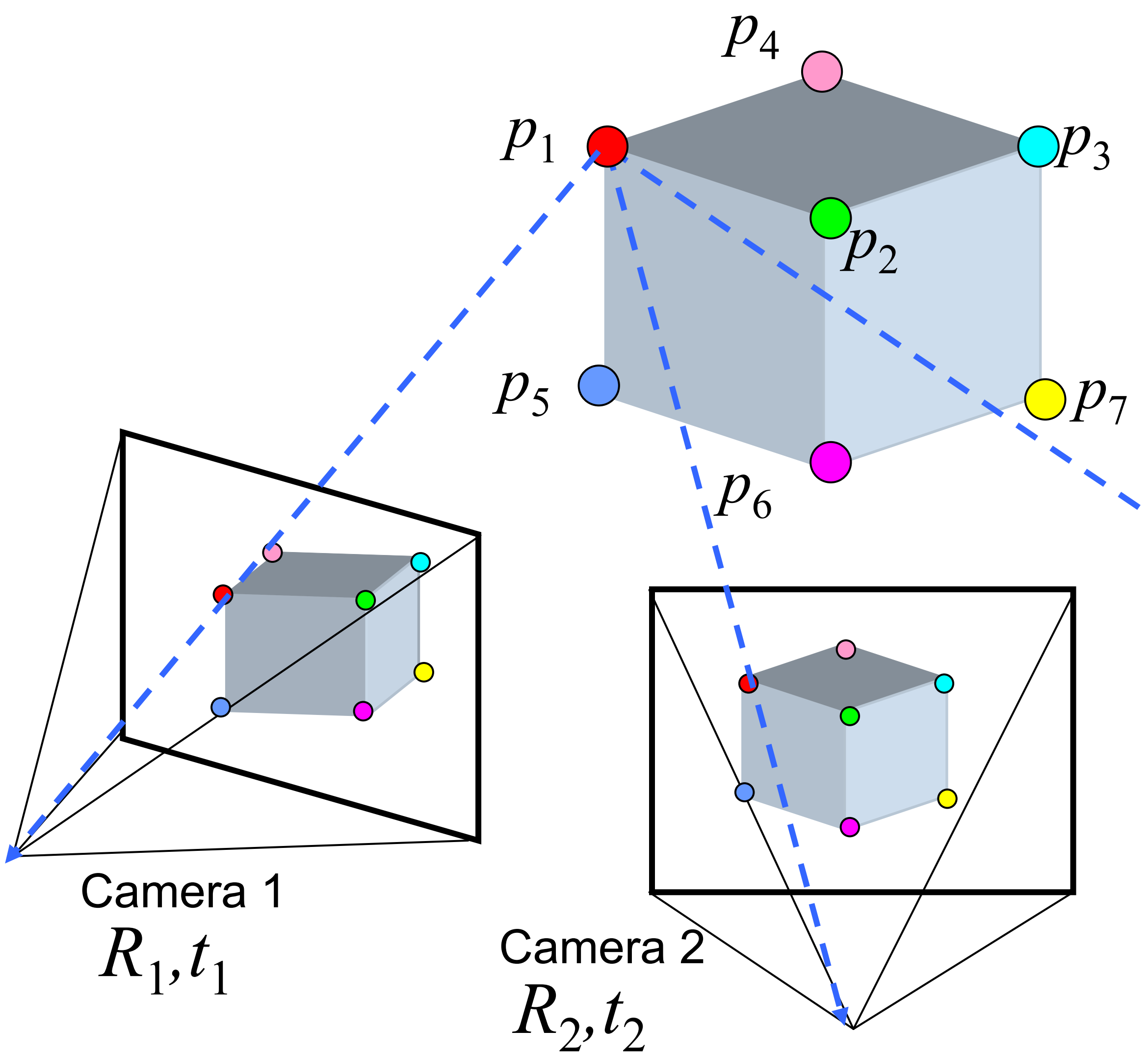


# Method – Step 5

1. Features/keypoint detection in the input images using *SIFT*
2. Features matching for each pair of images
3. Fundamental matrix estimation using the eight-point algorithm, using *RANSAC*
4. Removal of matches that are outliers to the estimated fundamental matrix
5. Structure from motion step to estimate the parameters of each pair of cameras, in a bundle adjustment process


cpalomo@inf.puc-rio.br

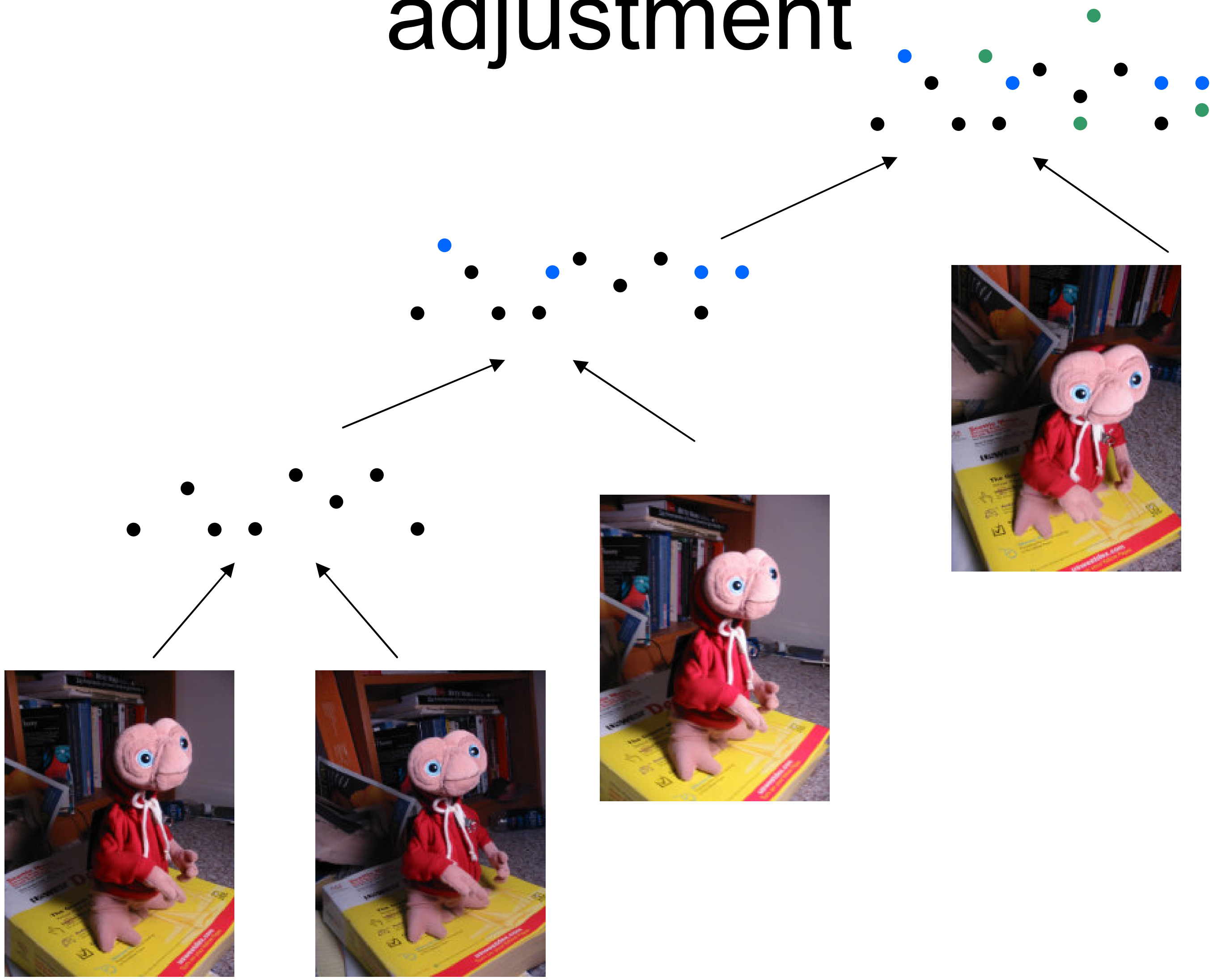
# Structure from motion (SfM)




*minimize*

$$\mathrm{f}\,(\mathbf{R}, \mathbf{T}, \mathbf{P})$$

# Incremental SfM: bundle adjustment



# Bundle adjustment code

- Noah Snavely and the University of Washington released their bundle adjustment code for non-commercial use
- Not just the bundle adjustment code, but all the SfM process is made by the released code
- After many (indeed) days sorting out problems with libraries compatibilities, made it work in Visual Studio

# Resulting framework

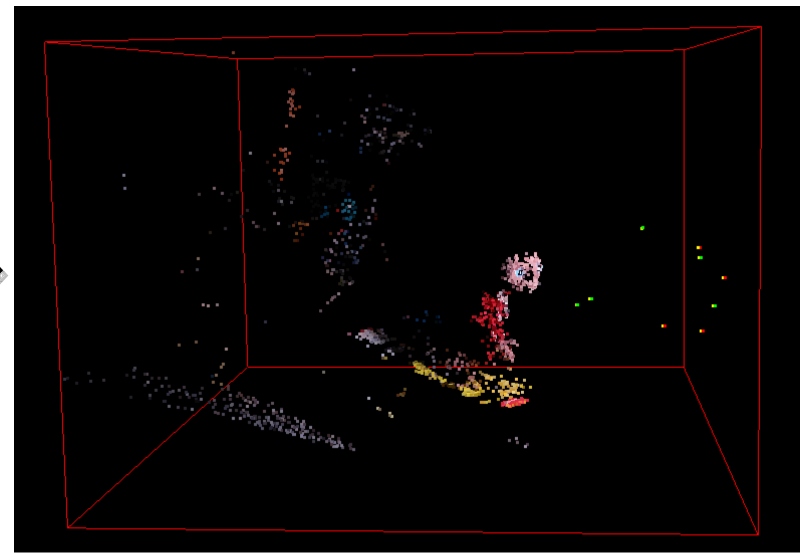
- Replaced code for features detection, matching and F-matrix estimation with my own, using only the SfM piece

**Features detection**

**Features matching**

**Robust F-matrix estimation**

**Incremental Sfm: bundle adjustment**

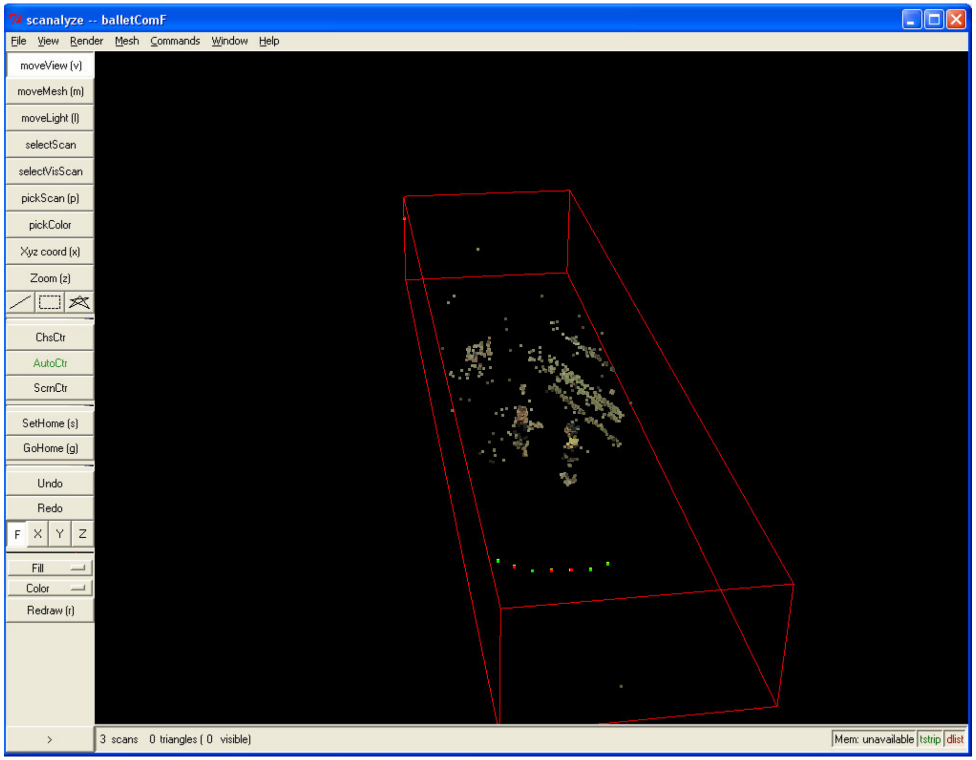


**Relative camera positions and orientations**

**Point cloud**

# Results

- Point cloud and relative camera positions and orientations can be viewed in Scanalyze (software by Stanford), since they are stored in a **ply** file



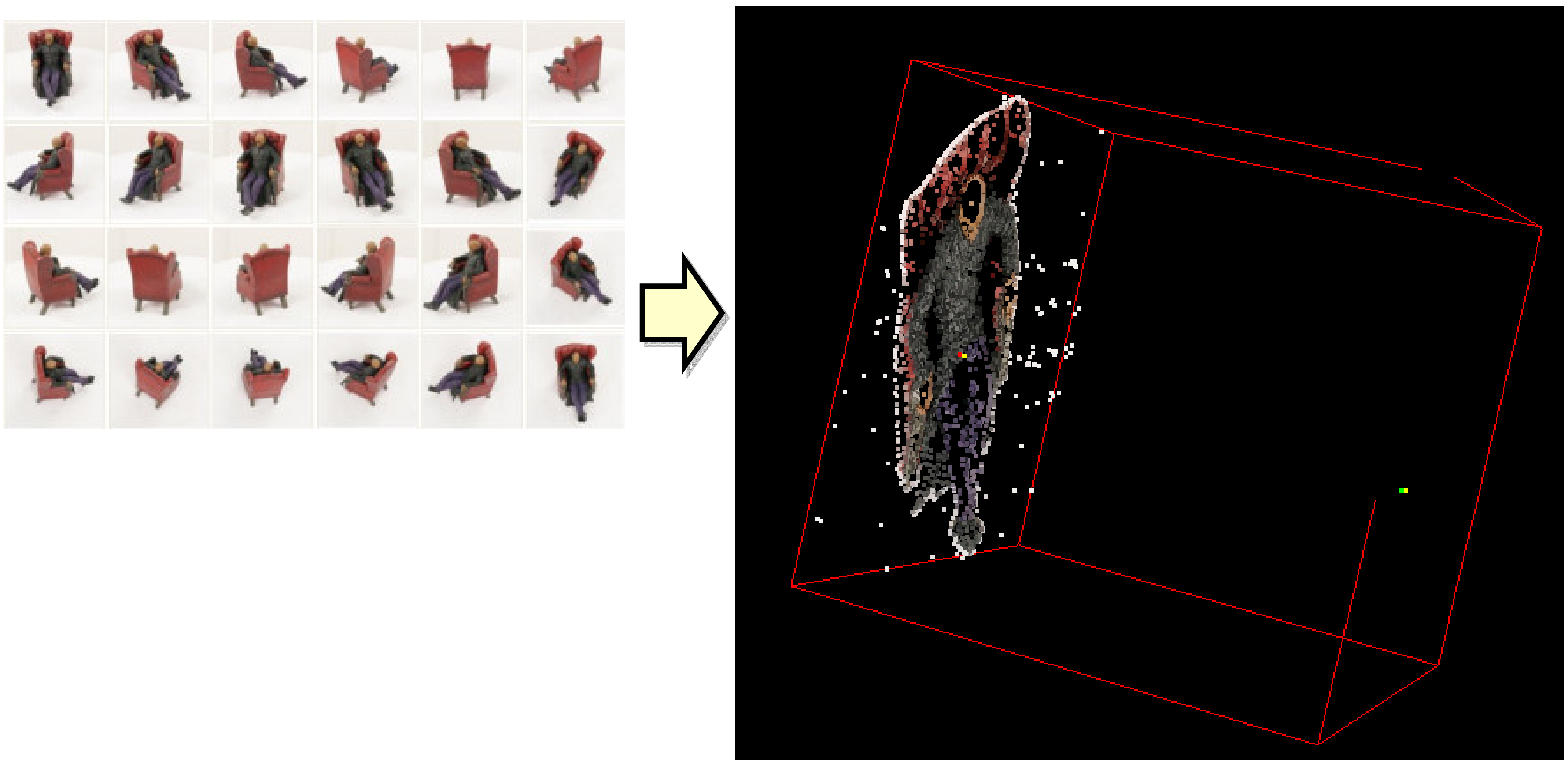
# Analysis of results

- The SfM results with our F-matrix estimate have been visually plausible, but further analysis must be done

- Since the bundle adjustment consists in a non-linear optimization (by use of Levenberg–Marquardt algorithm), it is prone to local-minima. Therefore, good initialization of parameters is necessary

- Especially the focal length of the camera used for the photos must be informed for the system, otherwise it may converges to a bad result
  - focal length is available through Exif tags for most consumer cameras nowadays

# Example of bad convergence to a local minima due to bad initialization of parameters



# Future work

- Make experiments with real Community Photo Collections to test the robustness and performance of the method
- Effectively evaluate the bundle adjustment method used, as well as the influence of the initial parameters on the final result
- Finish simple viewer: use IBR, fitting planes as impostors to project photos mixed with the registered points. Smooth transition between photos using blending.
- Experience with multi-view stereo using the output of this framework
- Experience with reconstruction from video
- Work on some contribution regarding the navigation process, as suggested by Luiz Velho: maybe mix IBR with planes as impostor for planar parts of a scene, like walls with paintings in a art gallery, and use full 3D reconstruction for statues

# References


- *Photo tourism: Exploring photo collections in 3D*
  Noah Snavely, Steven M. Seitz, Richard Szeliski.
  ACM Transactions on Graphics (SIGGRAPH Proceedings), 25(3), **2006**, 835-846

- *Finding Paths through the World's Photos*
  Noah Snavely, Rahul Garg, Steven M. Seitz, and Richard Szeliski.
  ACM Transactions on Graphics (SIGGRAPH Proceedings), 27(3), **2008**, 11-21

- *Multi-View Stereo for Community Photo Collections*
  Michael Goesele, Noah Snavely, Brian Curless, Hugues Hoppe, Steven M. Seitz
  Proceedings of ICCV 2007, Rio de Janeiro, Brasil, October 14-20, **2007**

- http://grail.cs.washington.edu/projects/cpc/

- http://www.cs.unc.edu/~ccwu/siftgpu/

- http://www.cs.umd.edu/~mount/ANN/

- *Multiple View Geometry in Computer Vision, second edition*
  Hartley, R.~I. and Zisserman, A.
  Cambridge University Press, ISBN: 0521540518, **2004**

# Thanks!

Thanks for the listeners of this talk!

Thanks must also be given to the University of Washington and Noah Snavely for making their SfM code available.

César Morais Palomo
cpalomo@inf.puc-rio.br
November 2008