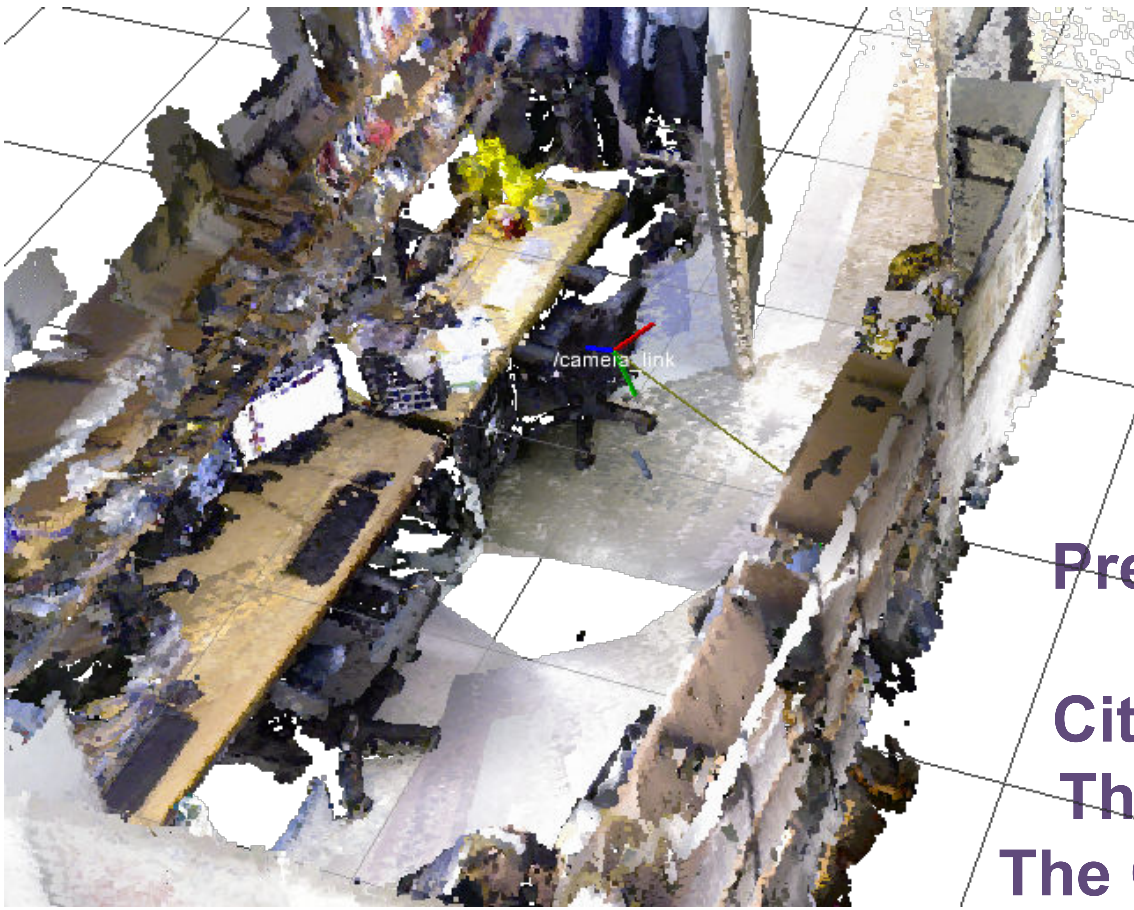
6-DoF Pose Localization in 3D Point-Cloud Dense Maps Using a Monocular Camera



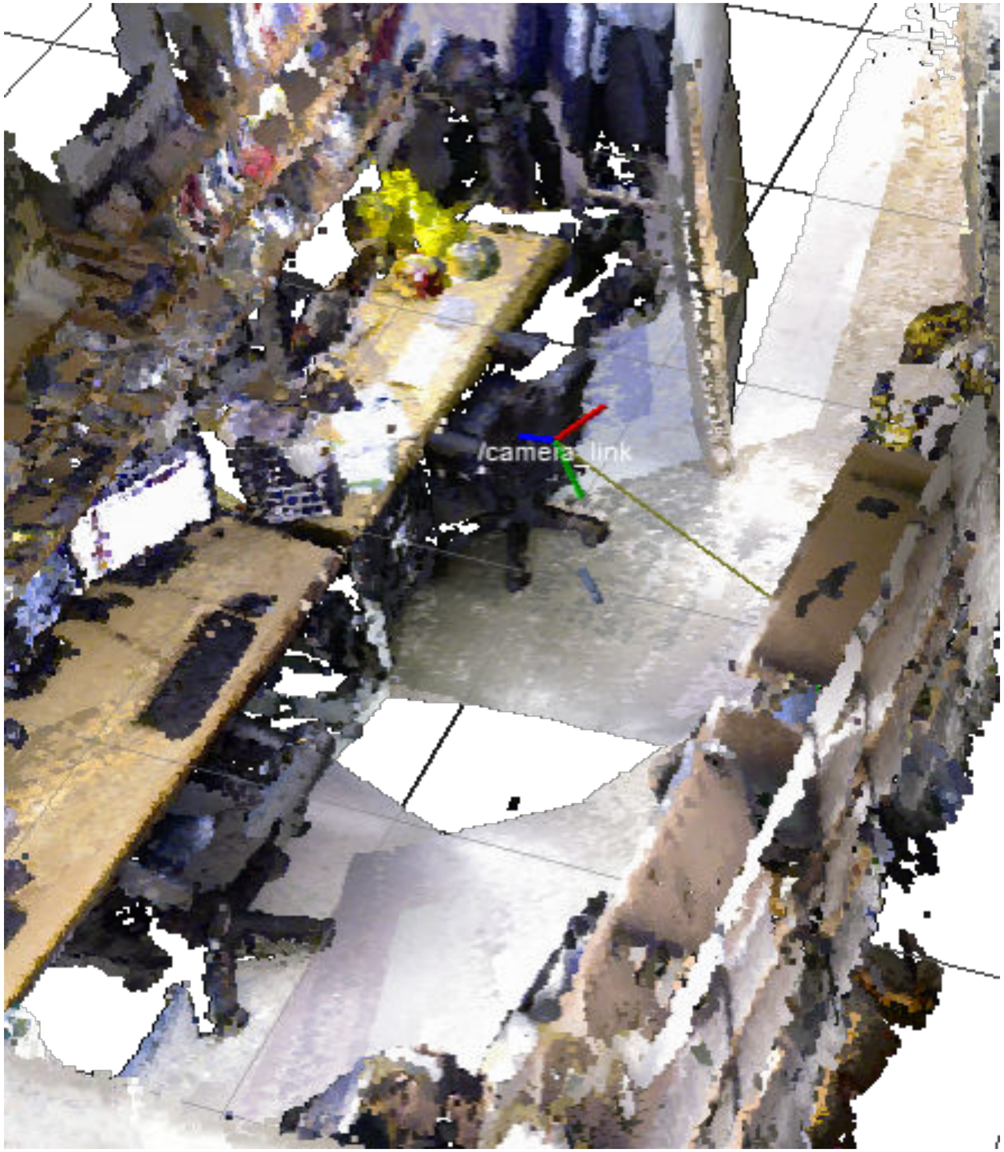
Authors:

Carlos Jaramillo[a]
Ivan Dryanovski[a]
Roberto Valenti[b]
Jizhong Xiao[b]

Presenter: Dr. Jizhong Xiao

City University of New York
The Graduate Center[a] and
The City College of New York[b]

Presentation Outline



1. **Problem description**
2. **Existing approaches**
   a) Monocular SLAM
   b) RGB-D SLAM
3. **Proposed method**
   a) Initial pose estimation
   b) System's pipeline
4. **Results**
   a) Experiments
   b) Performance
5. **Future work**

1. Problem Description

GOAL: 6-degree-of-freedom (6-DoF) pose localization

by simply using a monocular camera

+

inside a 3D point-cloud dense map

+

"prebuilt" with depth sensors

(e.g., RGB-D sensor, laser scanner, etc.)

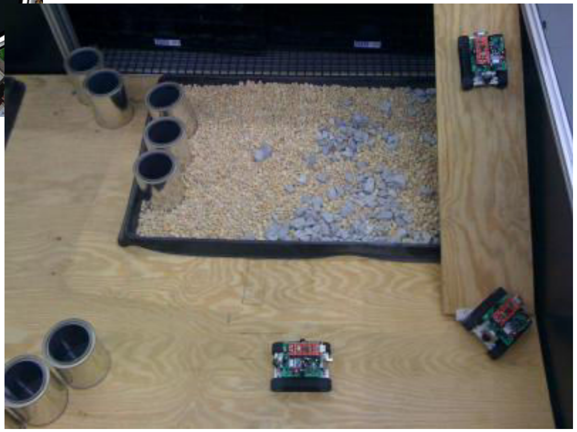
1. Problem Description

APPLICATION EXAMPLES: unconstrained motion of monocular cameras such as in smartphones or mounted in small robots

- **Augmented reality**
  - Showcases
  - Games
  - Museum tours

http://augmentedpixels.com

Jaramillo's DREU 2009



- **Mobile robot navigation**
  - Swarm navigation (Search and Rescue)
    1. A leader equipped with powerful sensor(s) creates a map
    2. Followers (with simple cameras) localize themselves in map

2. Existing approaches

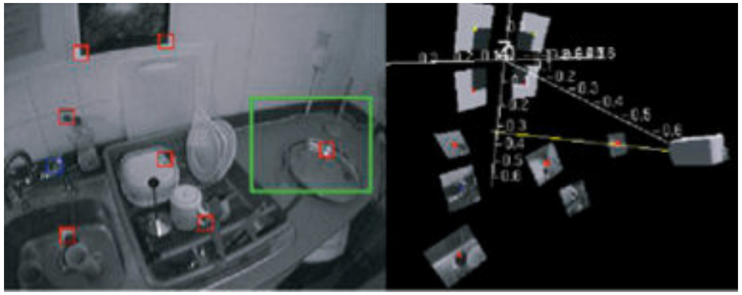
Visual SLAM: Visual *Simultaneous* Localization and Mapping

a) Monocular SLAM

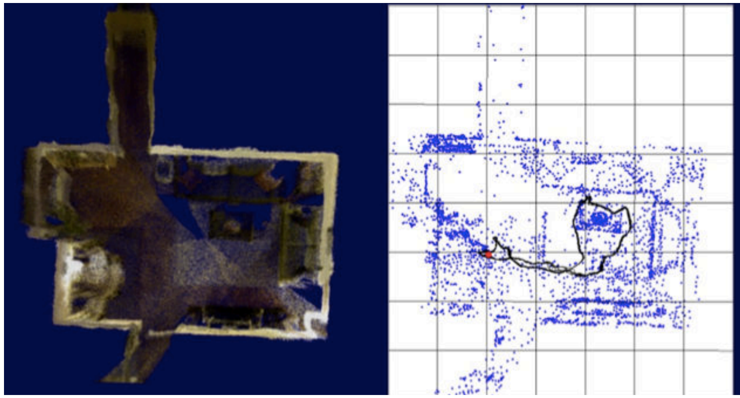
- MonoSLAM
  - » [2007, *Davison et. al.*]
- PTAM (Parallel Tracking and Mapping)
  - » [2007, *Williams et. al.*]
- Structure from motion (*Sfm*)
  - » [1981, Longuet-Higgins]

b) RGB-D SLAM

- Visual 3D SLAM
  - [2011, *Engelhard et. al.*]
- Fast 3D Mapping + Visual Odometry
  - [2013, *Dryanovski et. al.*]



Resource intensive:
Need to keep a **history of features** in the map



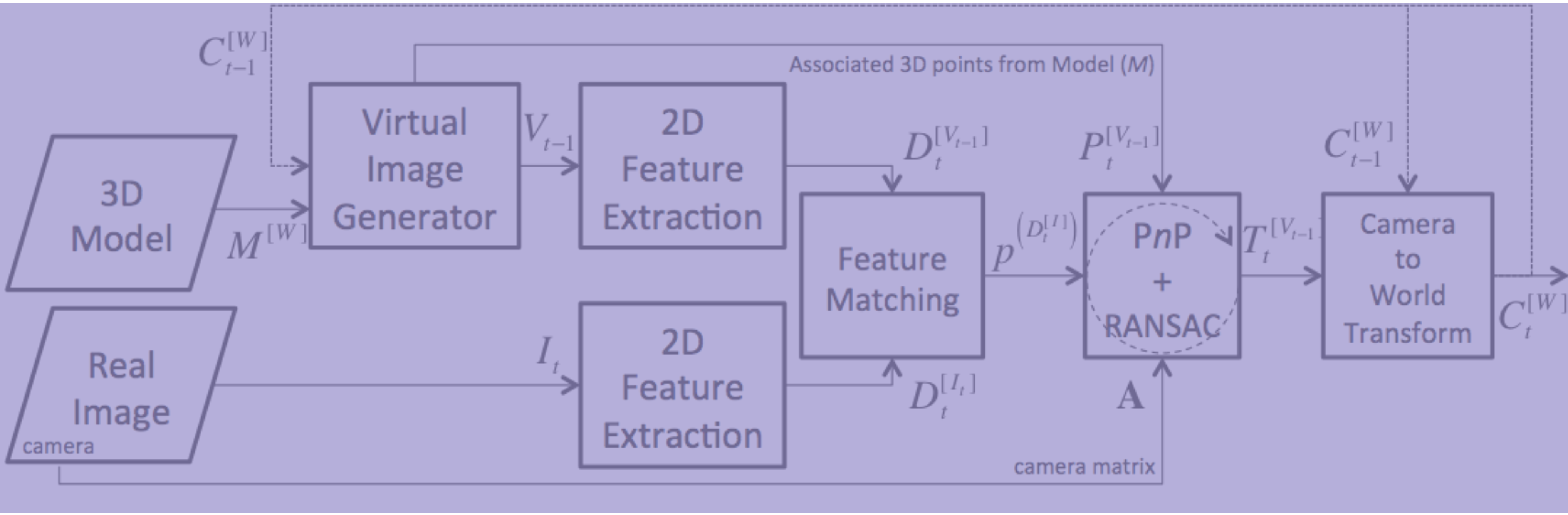
3. Proposed method

MONOCULAR LOCALIZATION WITHIN A 3D MAP

1. User initially maps out the scene (3D dense pointcloud)
   - Avoids resource-intensive Visual SLAM techniques
2. Our localization method:
   - Uses **dense** point-cloud (map)
   - Uses **single images** from a monocular camera
   - We **don’t track points**
   - We generate **virtual images** (using previous pose)

3. Proposed method

MONOCULAR LOCALIZATION WITHIN A 3D MAP (Pipeline)

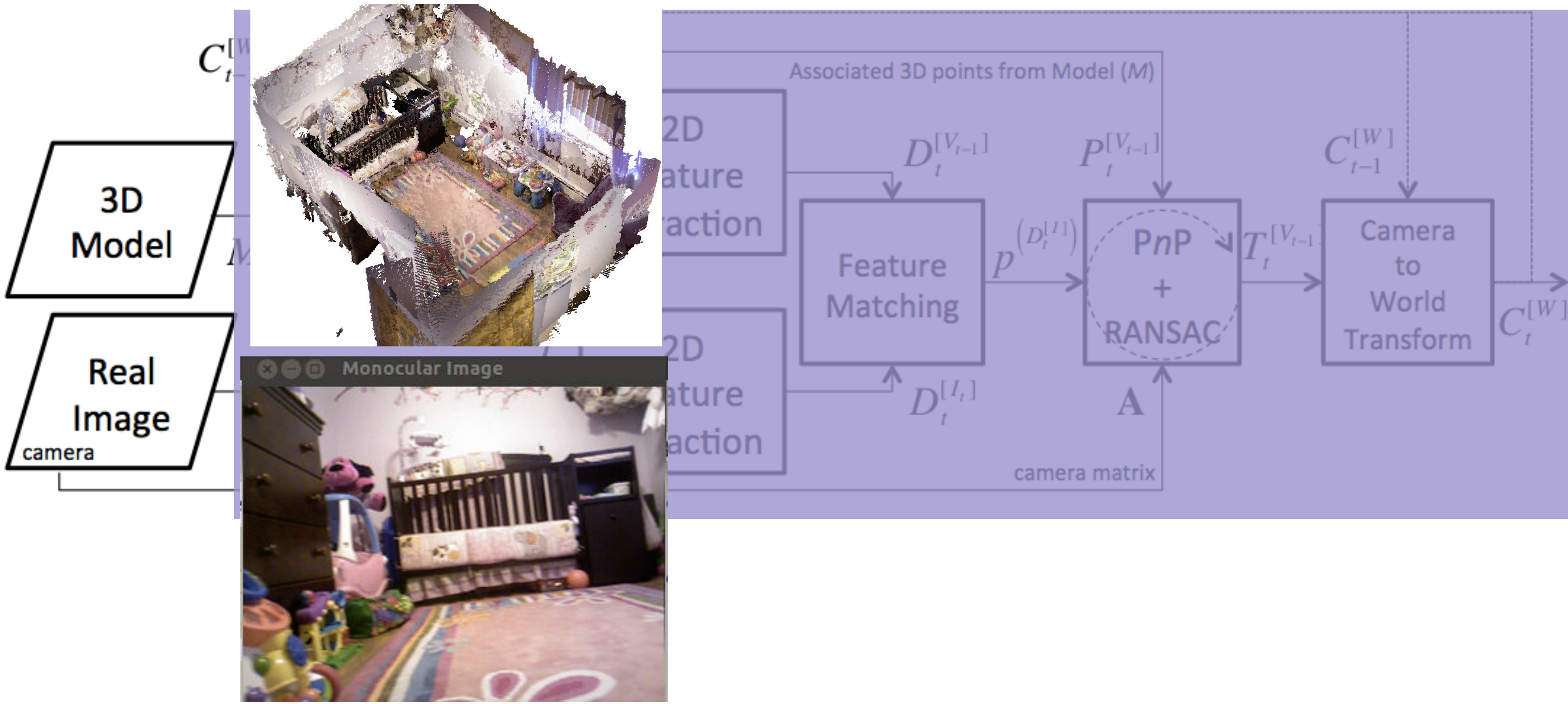


Initial pose estimation (first time only!):

1. In the first input image, I_1, we detect SURF. Also, extract SURF from all the *map's* frame images.
2. We train a descriptor matcher from all the SURF features.
3. For each feature in the real image, we find *n* nearest feature neighbors using the matcher.
4. Each feature in I_1 may point to a vector of descriptor matches. We take the top *n* candidates
5. The initial pose is found from a robust *PnP* matching between the *n* points from the real image and their corresponding 3D points in the map obtained from the top *n* matches.

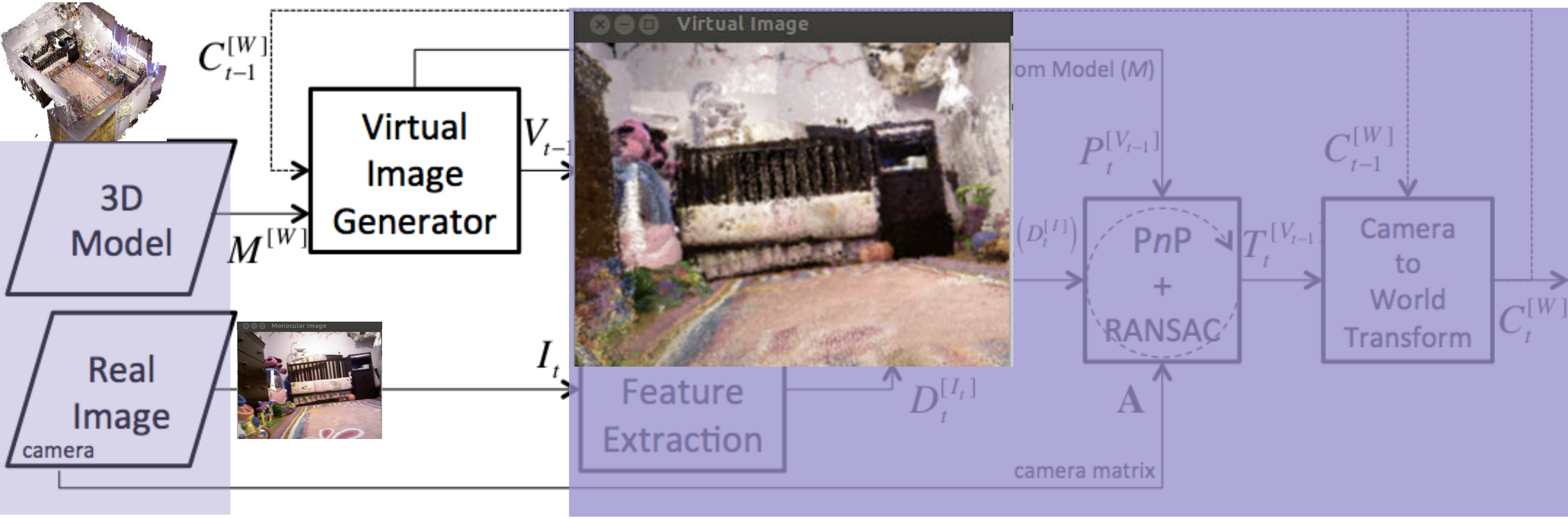
3. Proposed method

MONOCULAR LOCALIZATION WITHIN A 3D MAP (Pipeline)



3. Proposed method

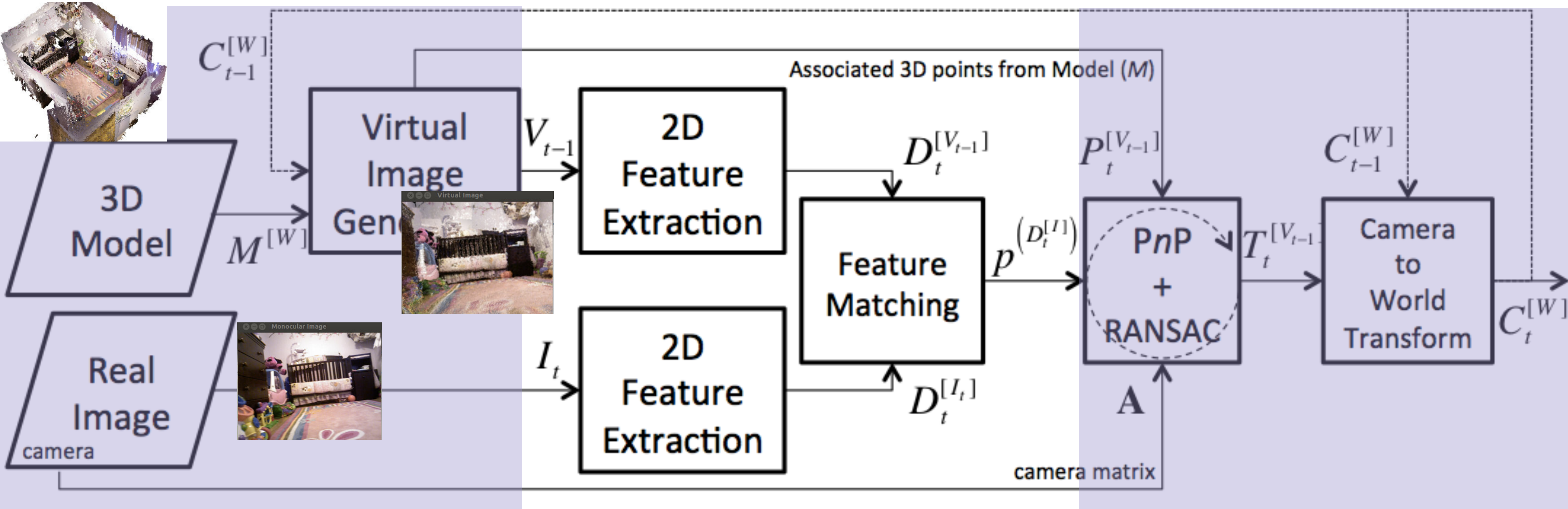
MONOCULAR LOCALIZATION WITHIN A 3D MAP (Pipeline)



1) The virtual view is constructed by projecting the map's 3D points to a plane using the t-1 pose.

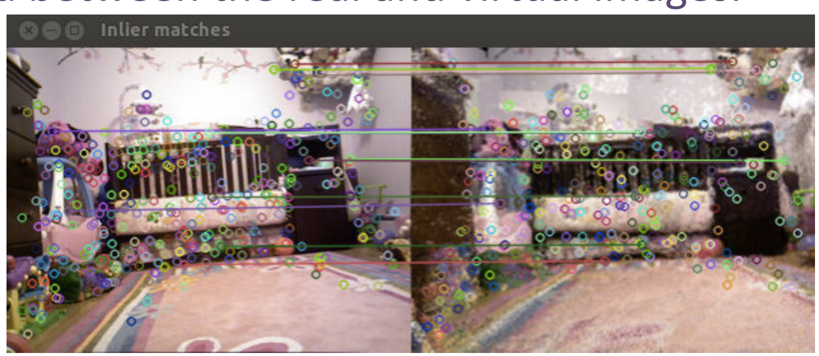
3. Proposed method

MONOCULAR LOCALIZATION WITHIN A 3D MAP (Pipeline)



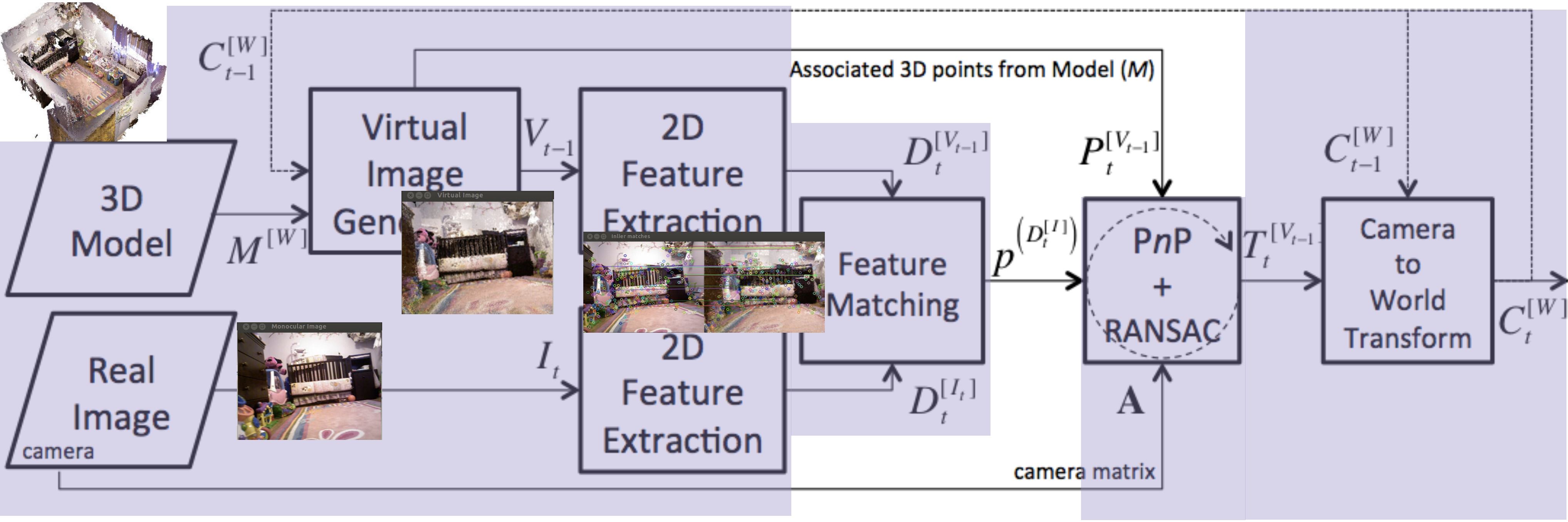
1) The virtual view is constructed by projecting the map's 3D points to a plane using the t-1 pose.

2) 2D features are matched between the real and virtual images.



3. Proposed method

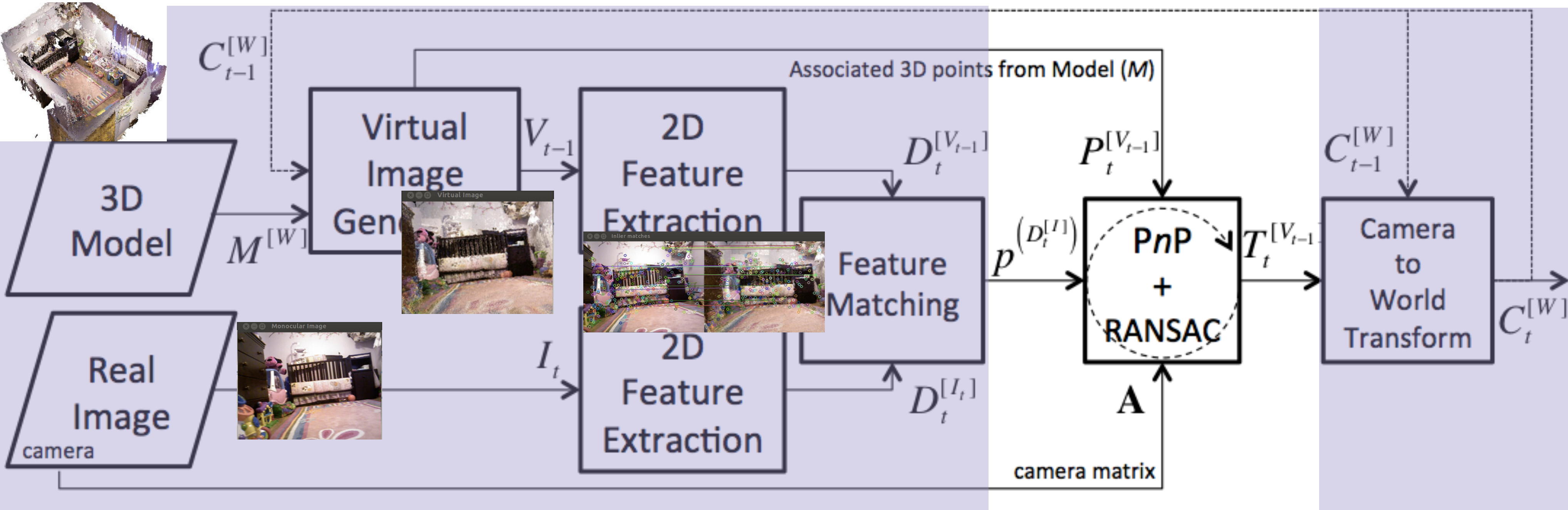
MONOCULAR LOCALIZATION WITHIN A 3D MAP (Pipeline)



1) The virtual view is constructed by projecting the map's 3D points to a plane using the t-1 pose.
2) 2D features are matched between the real and virtual images.
3) 2D-to-3D point correspondences are obtained between the real camera's 2D features and associated 3D points in the map.

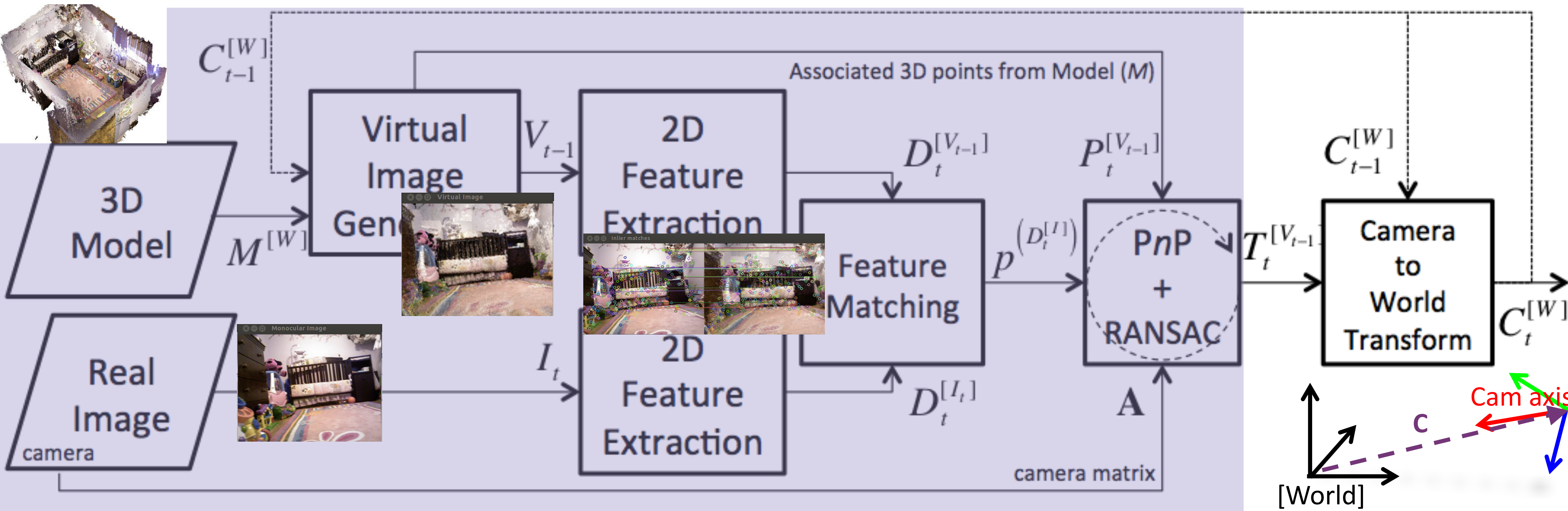
3. Proposed method

MONOCULAR LOCALIZATION WITHIN A 3D MAP (Pipeline)



1) The virtual view is constructed by projecting the map's 3D points to a plane using the t-1 pose.
2) 2D features are matched between the real and virtual images.
3) 2D-to-3D point correspondences are obtained between the real camera's 2D features and associated 3D points in the map.
4) After Perspective-n-Point (PnP) + RANSAC, the relative 6-DoF transformation between the real and virtual cameras is found.

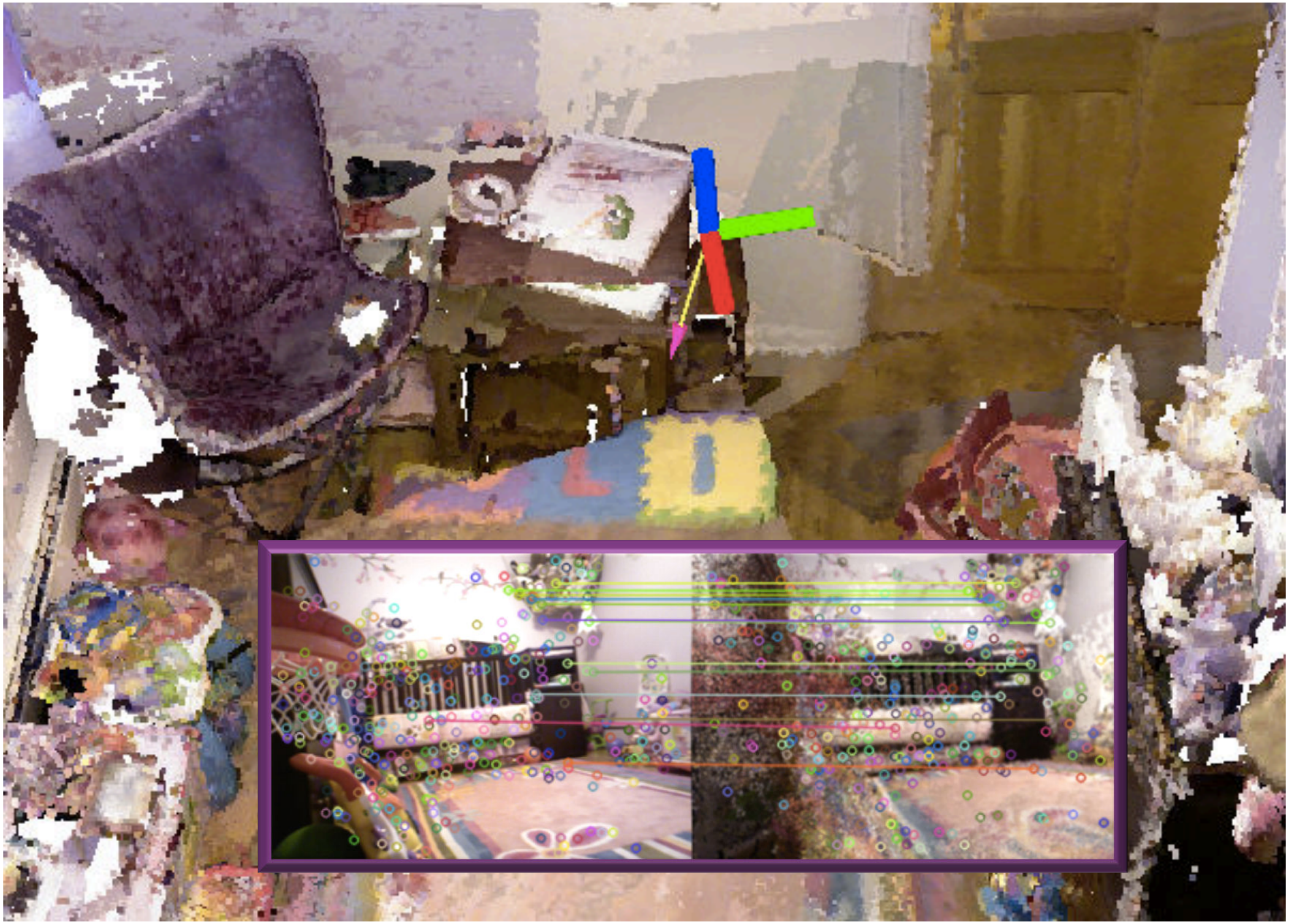
MONOCULAR LOCALIZATION WITHIN A 3D MAP (Pipeline)



1) The virtual view is constructed by projecting the map's 3D points to a plane using the t-1 pose.
2) 2D features are matched between the real and virtual images.
3) 2D-to-3D point correspondences are obtained between the real camera's 2D features and associated 3D points in the map.
4) After Perspective-n-Point (PnP) + RANSAC, the relative 6-DoF transformation between the real and virtual cameras is found.
5) A final frame transformation localizes the 6-DoF pose of the camera with respect to the map.

4. Results

Baby's room example (1)



4. Results

Baby's room example (2)

4. Results

Baby's room example (3)

4. Results

Baby's room example (4)

4. Results

Baby's room example (5)

4. Results

Office room example (Video)

GROVE SCHOOL OF ENGINEERING

6-DoF Pose Localization in 3D Point-Cloud Dense Maps Using a Monocular Camera

Authors:
Carlos Jaramillo
Ivan Dryanovski
Roberto Valenti
Dr. Jizhong Xiao

4. Results

- At **QVGA** resolution (320x240 pixels), the worst-case execution times running on a **1.7 GHz Intel Core i5** processor (inside a virtual machine) were:

Process (Per image frame)	Worst-case time (ms)
Virtual Image Generation	70
SURF feature detection and description	100
SURF Feature matching with FLANN	8
PnP with *RANSAC* (1000 iters, 50 inliers, 10 px reprj. error)	200
Total	**378**

- Bear in mind that these time values include the visualization overhead of the 3D map and the images.
- In the worst case, it can process 3 FPS

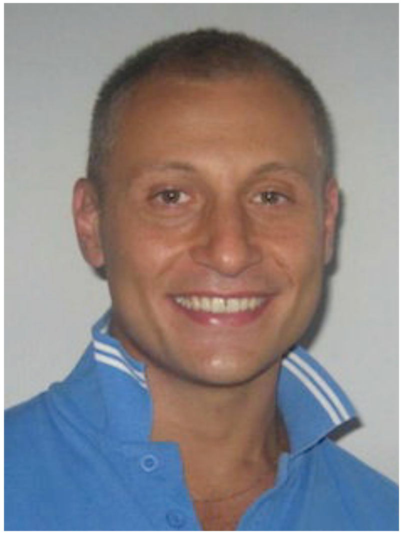
5. Discussion & Future Work

1. **Computing the initial pose of the camera adds an initial delay before the live image-feed can enter the pipeline.**
2. **We must improve quality of the virtual images**
   – Affects the feature correspondence procedure.
3. **Improve quality of 3D maps**
   – Virtual images depend on model density (Try meshed models)
4. **We have to validate our method by experimenting with bigger maps**
5. **We have to performing error analysis with ground truth data sets.**
   – Existing data sets don't produce dense maps
6. **Other enhancements:**
   1. Aid the rotation estimation with IMU sensors (phones have it)
   2. Use wider field-of-view real (and virtual) images in order to tolerate drastic motion.
   3. Support dynamic environments (only static environments today).

Thank you!

Jaramillo, Carlos

Dryanovski, Ivan



Valenti, Roberto

Xiao, Jizhong