

Sensor-enabled Intelligent Environments

Nico Blodow, Mihai Dolha, Zoltan-Csaba Marton,
Dejan Pangercic@IAS

7. Mai 2010





Today's Roadmap

- ▶ Sensors
- ▶ Data, Messages
- ▶ Calibration
- ▶ Point Cloud Processing
- ▶ ROS
- ▶ TF
- ▶ Visualization
- ▶ HW and Q&A



Sensors

Sonar sensors



same principles as used by bats, dolphins, RADAR and the police...



Sensors

Imaging Sensors



- ▶ cameras (monocular, black/white or color, distorted or rectified)
- ▶ stereo cameras (two units or complete stereo assemblies):
 - left: stereo rig using 2 regular cameras
 - right: dedicated stereo camera with on board processing (STereo On a Chip - STOC)
- ▶ depth information is extracted by correlating feature points in both images, and using the camera rig geometry (transformation between sensor origins)
- ▶ other cameras possible (thermal, ...)



Sensors

Imaging Sensors

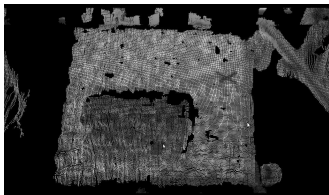
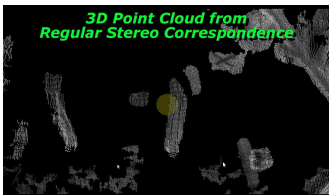


- ▶ Problem with stereo cameras: needs to match points in left and right image
- ▶ Textureless objects (e.g. white wall) do not get picked up
→ *sparse* depth image
- ▶ Possible solution: Injecting artificial texture by projecting a light pattern onto scene



Sensors

Imaging Sensors



- ▶ Problem with stereo cameras: needs to match points in left and right image
- ▶ Textureless objects (e.g. white wall) do not get picked up → *sparse* depth image
- ▶ Possible solution: Injecting artificial texture by projecting a light pattern onto scene



Sensors

3D Acquisition Devices

time of flight based:



- ▶ principle: send beam of light, beam hits target, measure time between sending the beam and receiving the back scatter
- ▶ if round trip time is t , then distance is $(ct)/2$.
- ▶ Often, phase difference is used instead of actual time of flight measurement.
- ▶ Use rotating mirror to deflect beam → 2D Scanner



Sensors

3D Acquisition Devices

time of flight based:



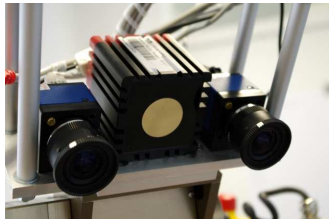
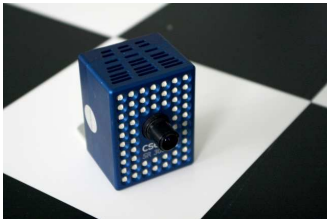
- ▶ **tilting** the plane of the laser beams results in a 3D acquisition device
- ▶ usually, 2D scanners are moved as a whole, but there are some "real" 3D scanners
- ▶ Problem: usually slow



Sensors

3D Acquisition Devices

time of flight cameras:



The phase difference principle can be exploited to directly measure a depth image:

- ▶ The light is coming from an array of LED's, pulsed at a certain frequency
- ▶ A special CCD sensor is used that measures intensity AND phase difference for each pixel
- ▶ Problems: "cross talk" between image regions



Sensors

3D Acquisition Devices

triangulation based:



(image courtesy of www.david-laserscanner.com)

- ▶ Laser line is swept over object, camera is picking up the resulting image.
- ▶ Transformation between camera and light source is known through hardware setup or through in-image features, e.g. how the line falls onto known background



Data, Messages

Point Clouds



left: unstructured point cloud

right: since points were measured in a grid like fashion, one can naively triangulate the scan (reflectance intensity viewed in color)



Data, Messages

Point Clouds

ROS message type:

```
user@hostname:~$ rosmmsg show sensor_msgs/PointCloud
Header header
  uint32 seq
  time stamp
  string frame id
geometry_msgs/Point32[] points
  float32 x
  float32 y
  float32 z
sensor_msgs/ChannelFloat32[] channels
  string name
  float32[] values
```

Basically, a vector of Point32, and a list of channels
example channels: normal vectors, intensity, color (r,g,b),
curvature...



Data, Messages

Surface Mesh

Some examples of objects:





Data, Messages

Surface Mesh

```
user@hostname:~$ rosmmsg show TriangleMesh
[triangle_mesh/TriangleMesh]:
Header header
  uint32 seq
  time stamp
  string frame_id
geometry_msgs/Point32[] points
  float32 x
  float32 y
  float32 z
geometry_msgs/Point32[] normals
  float32 x
  float32 y
  float32 z
triangle_mesh/Triangle[] triangles
  int32 i
  int32 j
  int32 k
string sending_node
```

Again, a vector of Point32, this time with vector of normals and a list of triangles

triangle list is a vector of triplets of indices into the points vector



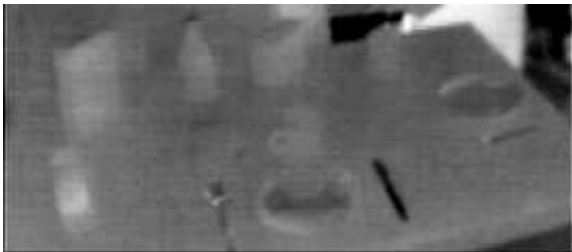
Overview

- ▶ Calibration is the transformation of a measurement to represent the measured value as closely as possible
- ▶ Sensors have non-linearities, varying fabrication errors, synchronization problems, and are influenced by external factors that are hard to eliminate
- ▶ Examples: photometric camera calibration, monocular camera calibration (resectioning/undistortion), calibrating a stereo rig, calibrating time-of-flight cameras, and calibrating 3D sensors with cameras.



Single Camera

- ▶ Pixel colors don't correspond to real colors (depends on lighting conditions) + reflections
- ▶ In a fixed/known environment you can deal with it automatically (not in service robotics), ex: histogram matching
- ▶ White balance can mess things up

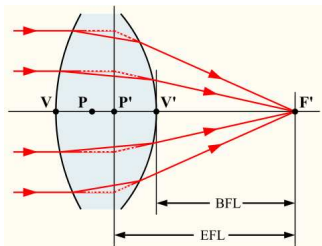


FLIR thermal camera



Single Camera

- ▶ Identify optical center, focal point, distortions by using a known pattern and non-linear minimization
- ▶ http://www.ros.org/wiki/camera_calibration/Tutorials/MonocularCalibration
- ▶ http://www.ros.org/wiki/camera_calibration/Tutorials/StereoCalibration
- ▶ http://www.ros.org/wiki/image_pipeline/CameraInfo (http://www.ros.org/doc/api/sensor_msgs/html/msg/CameraInfo.html)





Single Camera

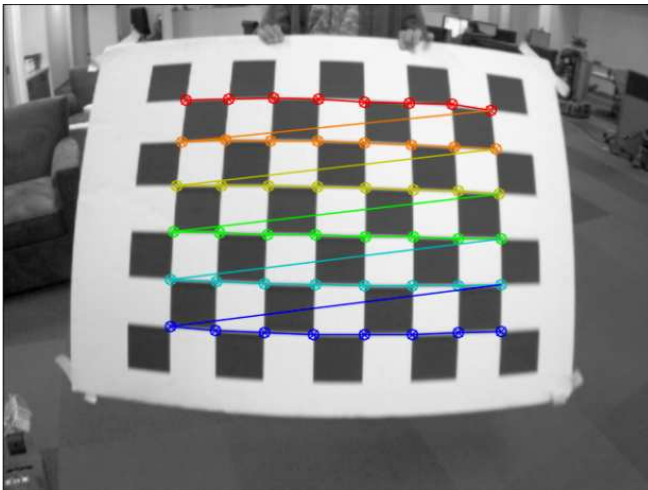
- ▶ If pin-hole camera model, and only skew is assumed: intrinsic calibration is a linear system with 5 unknowns
- ▶ Result can initialize better methods that minimize backprojection error



www.ros.org



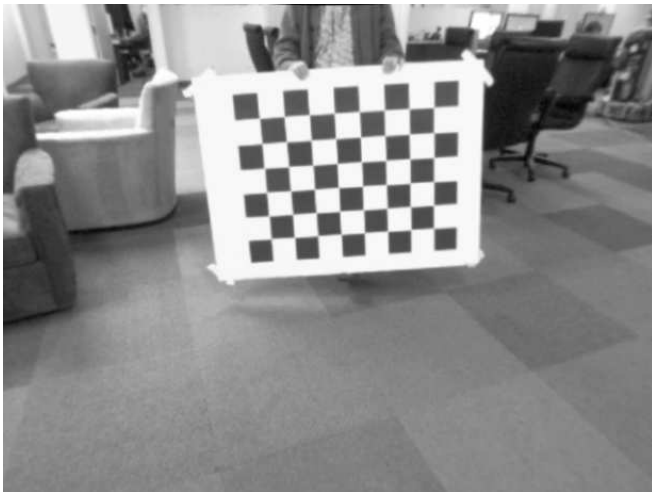
Single Camera



www.ros.org



Single Camera



www.ros.org



Stereo Camera

- ▶ Find the relative poses of two cameras, to be used for approximating depth information

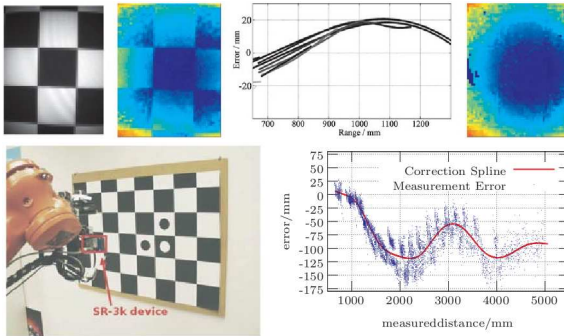


www.ros.org



3D Camera - Time-Of-Flight

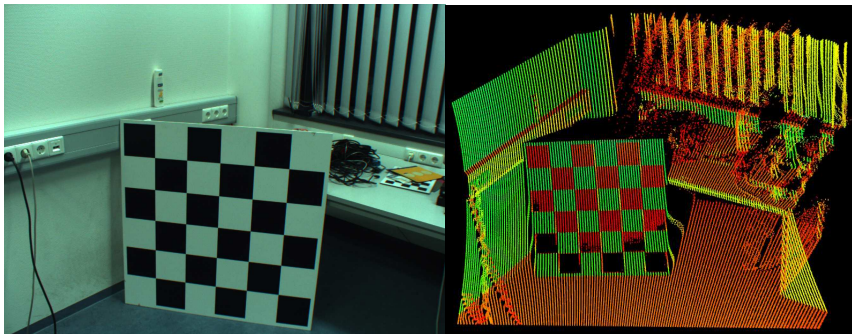
- ▶ TOF measurements are highly influenced by reflectivity
- ▶ Very hard to de-correlate things, you fix a problem and introduce a new one
- ▶ SwissRanger 4000 has already almost straight planes under the right conditions :)





Calibrating 3D Sensors with Cameras

- ▶ Where is the optical center of a tilting laser?
- ▶ What if something is seen only in one sensor?

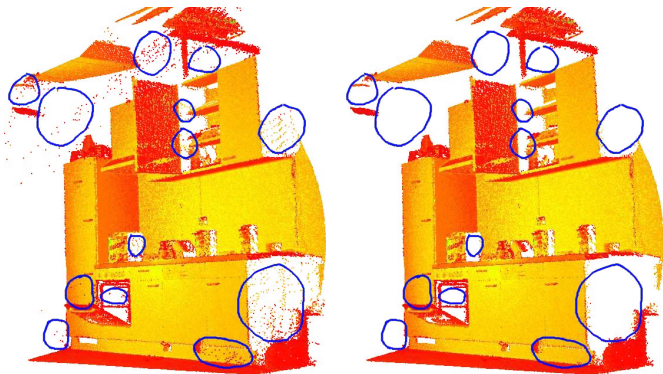


Camera image (left) and laser scan (right) of a checkerboard.
The 3D points were colored using the intensity values and “photographed” using a virtual camera model.



Point Cloud Processing

Removing Artifacts



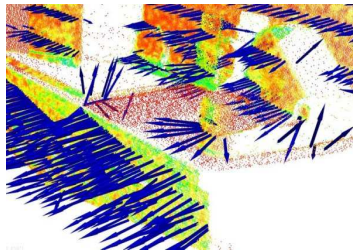
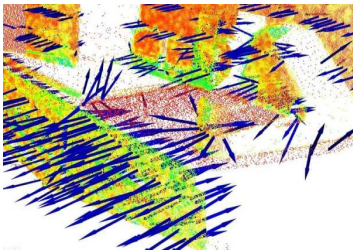
left: scan with artifacts before sparse outlier removal

right: artifacts got removed by iterating through points and removing those with very low point density



Point Cloud Processing

Normal Estimation



left: after normal estimation, normal vectors are generally randomly oriented

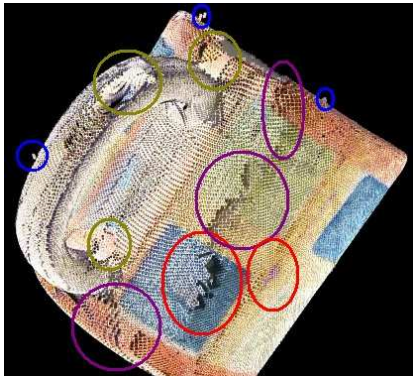
right: propagating normal orientation consistency (flip normals to view point or more complex methods)



Point Cloud Processing

Resampling/Smoothing and Triangulation

- ▶ Approximating the surface locally using a polynomial and/or a triangle mesh
- ▶ To correct/downsample the raw data and transform the point based representation into a 3D surface

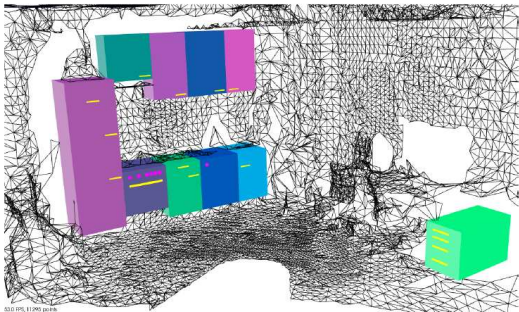
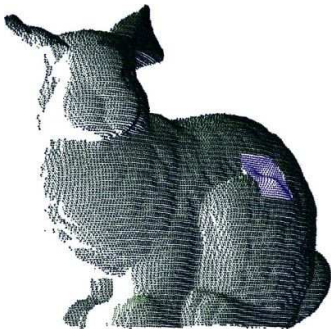




Point Cloud Processing

Resampling/Smoothing and Triangulation

- ▶ Approximating the surface locally using a polynomial and/or a triangle mesh
- ▶ To correct/downsample the raw data and transform the point based representation into a 3D surface





- ▶ `roscd pkg_name` : change directory to package
- ▶ `rostopic` : list topics or echo a specific one
- ▶ `rosmmsg` : lookup a message type
- ▶ `roslaunch` : launch a collection of nodes from an XML-based "launch file"
- ▶ `rosparam` : access the parameter server (central location to set and retrieve parameters for nodes)

familiarize yourself with them, they will save a lot of time



TF

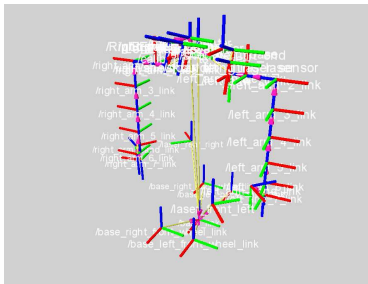
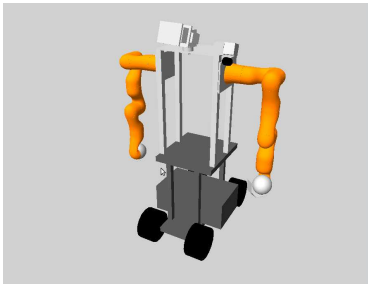
Transform Library

- ▶ transform: translation (x,y,z) and orientation (roll, pitch, yaw angles, encoded in quaternion)
- ▶ on topic **tf**, there is a tree of transforms
- ▶ every sensor has it's own coordinate frame, and publishes data in this **frame_id**



TF

Transform Library Continued



left: visualization model of our KUKA-based robot (Rosie)
right: corresponding tf-tree



TF

Transform Library Continued

- ▶ a robot description file stores the hierarchy of transforms about the robot
- ▶ if a consumer of sensor data needs the data in another transform, it can request the corresponding transform (between `data frame_id` and `target frame_id`) from a `tf::TransformListener` Instance
- ▶ this query also states a certain time (since the frames can be moving, e.g. tilting laser)
- ▶ `tf` automatically can interpolate in time
- ▶ Advantages: great way of modularizing, since only the robot description states the hierarchy, no producer or consumer of data needs to care



TF

Transform Library Continued

```
user@hostname:~$ rostopic echo tf
---
transforms: [
  header: PG
    seq: 0
    stamp: 1273224129201106407
    frame_id: /parent_frame
    child_frame_id: /child_frame
  transform:
    translation:
      x: 1.0
      y: 0.0
      z: 0.0
    rotation:
      x: 0.0
      y: 0.0
      z: 0.850903524534
      w: 0.525321988818]
```



Visualization

rviz

Alt-TAB to live demo :P



Homework

- ▶ see course's website

- ▶ Questions?