Projektgruppe FastSense

Meilenstein 2

Hardware Accelerated TSDF SLAM

28. Oktober 2020

Inhalt

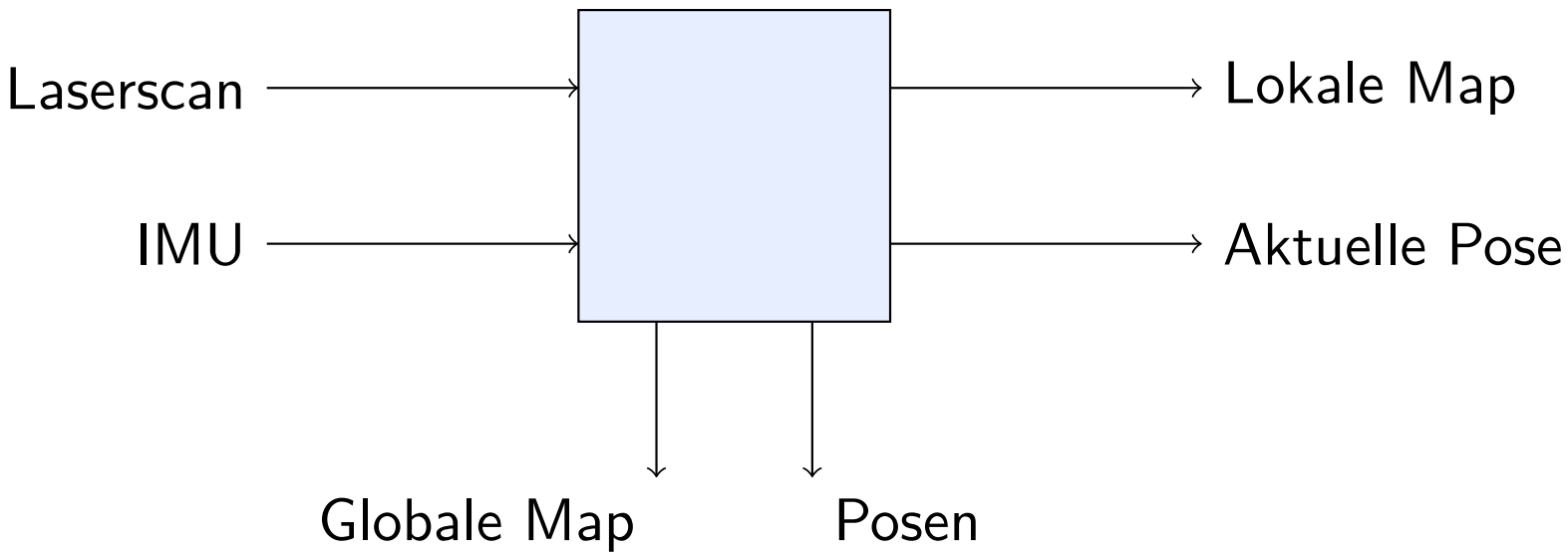
Ziele und Anforderungen

Ziele für MS2

- Implementation von inkrementellem TSDF SLAM in „autarker“ Box
- Vorimplementation in Software
- Implementation einzelner Komponenten in Hardware
- Speicherung von Pose-Graph und TSDF-Karte zur Rekonstruktion des kompletten explorierten Bereichs
- Evaluation durch Zeit- und Strommessung

Funktionale Anforderungen

- Lokale TSDF-Map ausgeben
- Aktuelle 6D-Pose ausgeben
- Map auf Basis der IMU und Velodyne-Daten
- Trajektorie und TDSF-Map für jede Pose speichern
- Parameter zur Laufzeit anpassbar



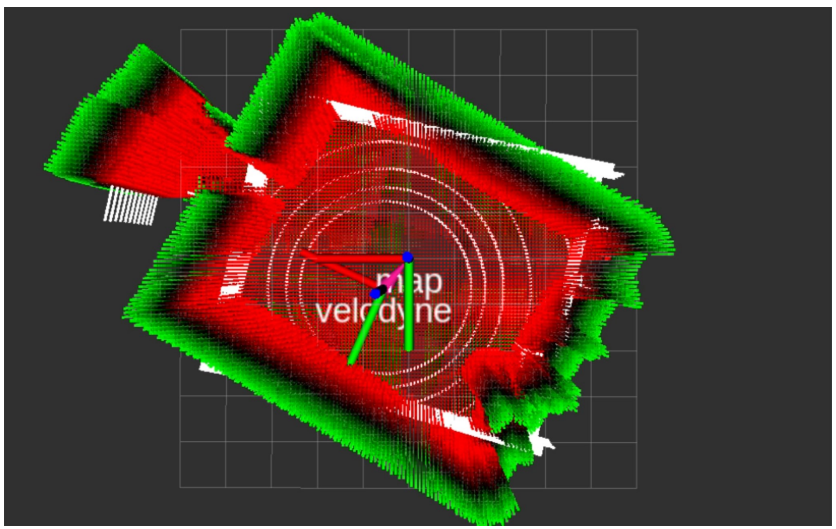
Nicht-Funktionale Anforderungen

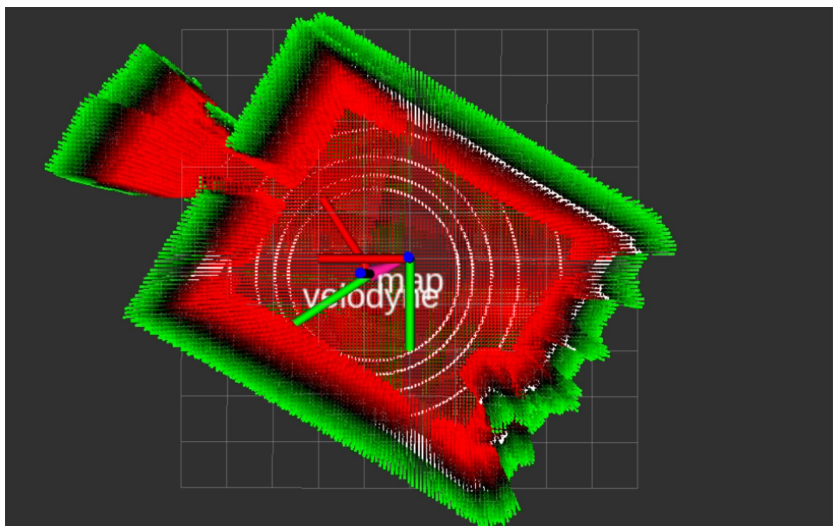
- HW-Plattform: Trenz-Board
- Entwicklungsplattform: Vitis
- Beschleunigung der Algorithmen durch FPGA
- Sensoren direkt am Board
- Unit-Tests
- Testbench
  - Integration, Strommessung, Zeitmessung, Visualisierung
- Logging

Implementierung

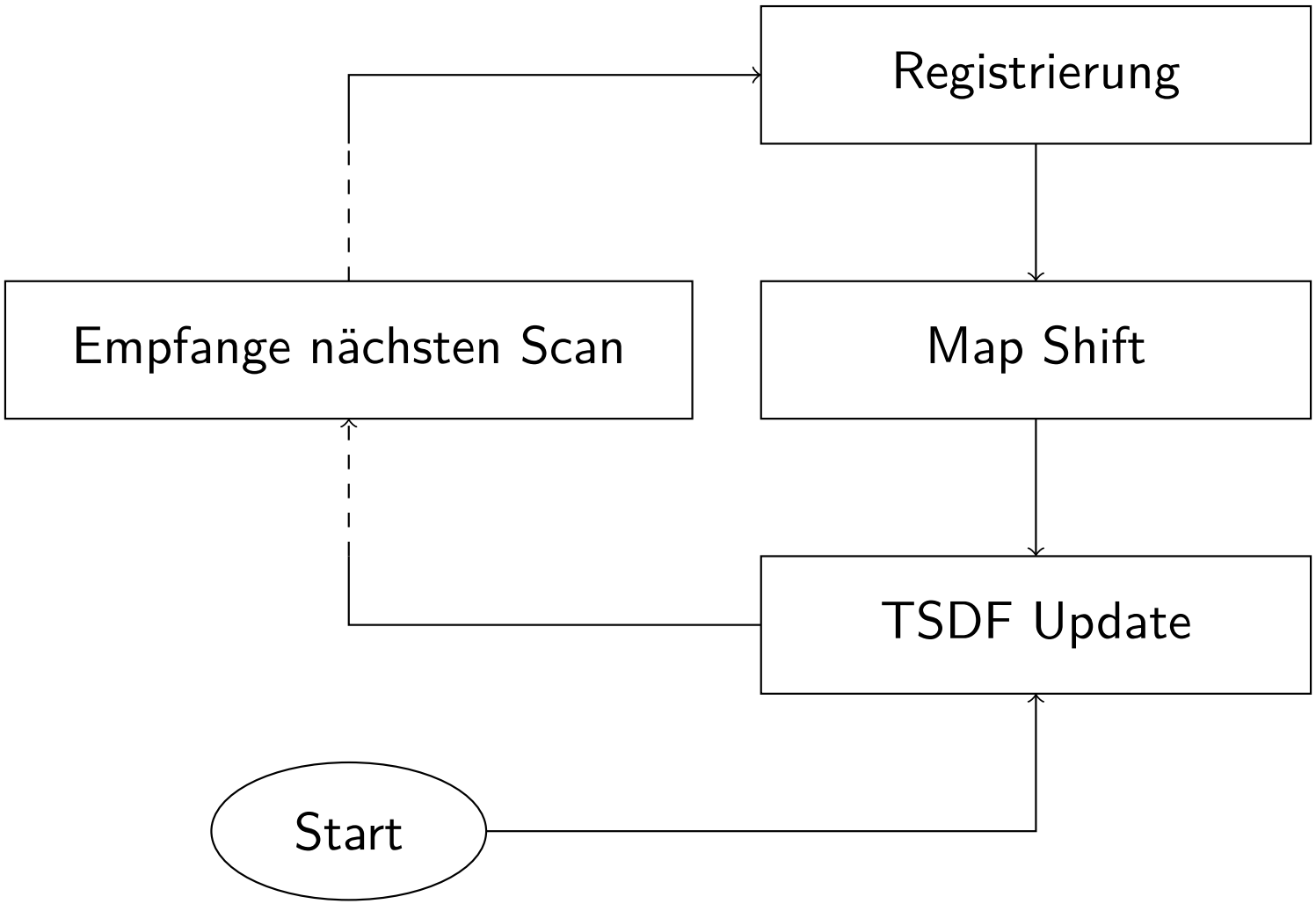
Recap: Prototyping Demo

- Gute Parameterkombination herausgefunden
- Geplante Funktionalität war vorhanden und in RViz darstellbar
- Erkannte Probleme / Bottlenecks:
  - Laufzeit stark abhängig von der Auflösung der Karte
  - Probleme mit Orientierung (kurz nach Demo gefixt)
  - Insgesamt noch recht langsam (∼0.5s/Scan auf Glumanda, 2-5s/Scan in Testwelt)
  - Registrierung und TSDF-Update als parallelisierbare Bottlenecks

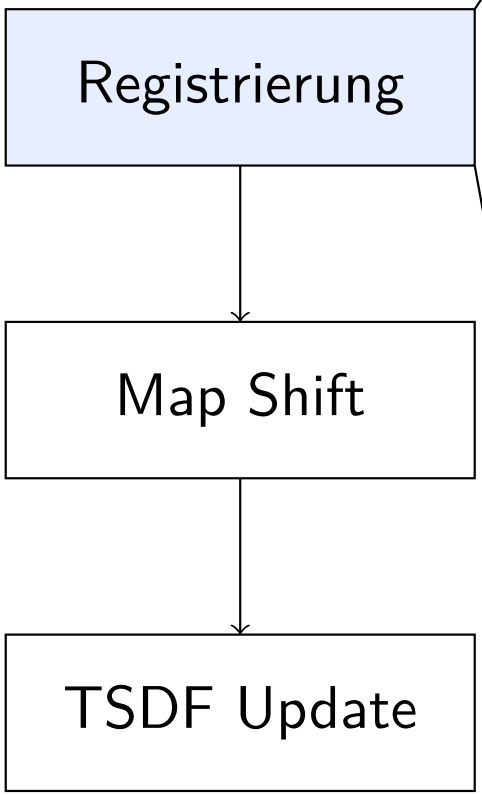




Algorithmus



Algorithmus



T = Initial Transform;

Transform point cloud with T;

do

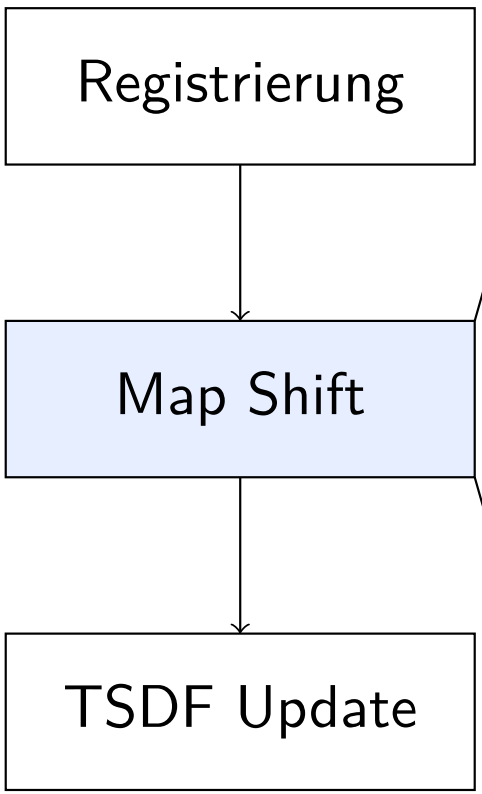
  E = Minimization of the sum of signed distances squared;

  Transform point cloud with $T(E)$;

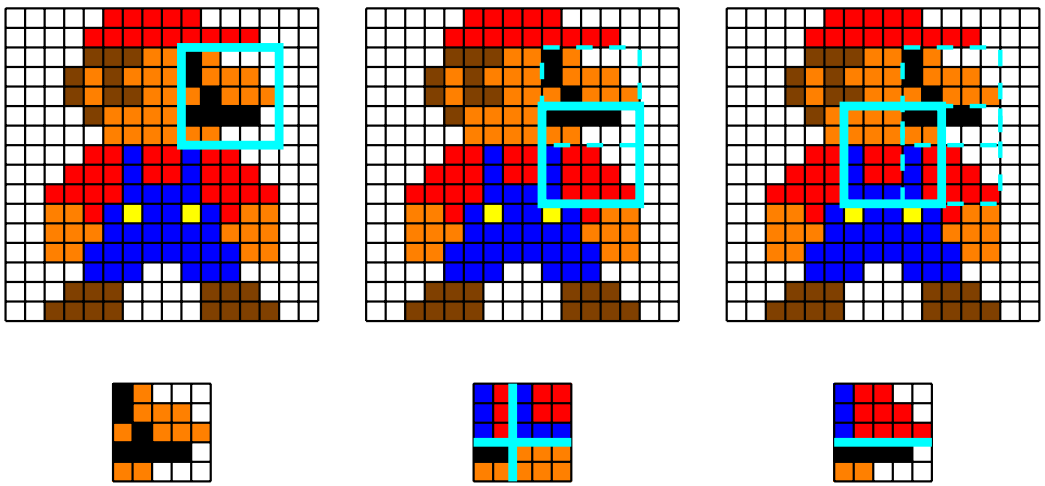
while *under max iterations and E not converged*;

return T;

Algorithmus

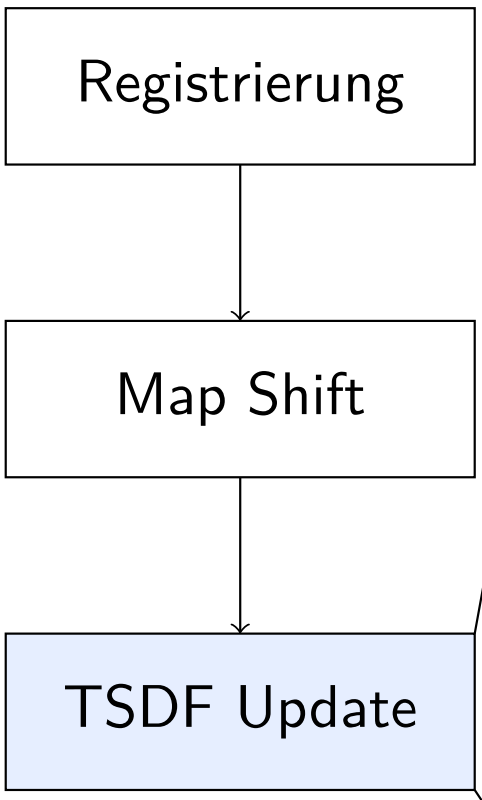


- LocalMap (aka RingBuffer)
  - Geteilt zwischen SW und HW

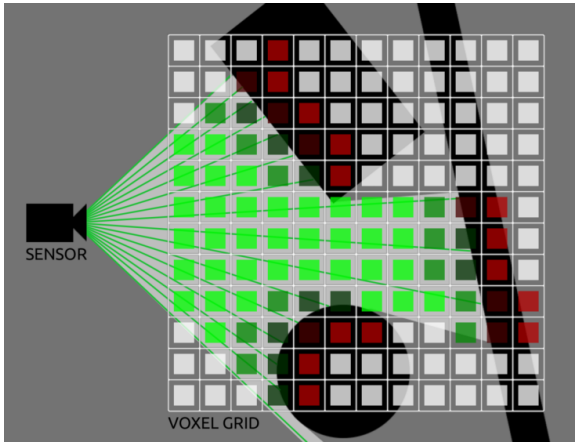


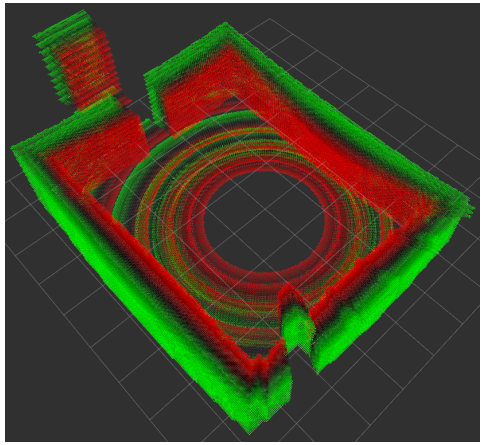
- Werte außerhalb in GlobalMap
  - Chunk-basiert
- Speicherung in HDF5

Algorithmus



1. TSDF Generierung





[1]

2. Update mit gewichtetem Mittelwert

$$M_V = \frac{M_V \cdot M_W + v \cdot w}{M_W + w}$$

$$M_W = \min(M_W + w, W_{\max})$$

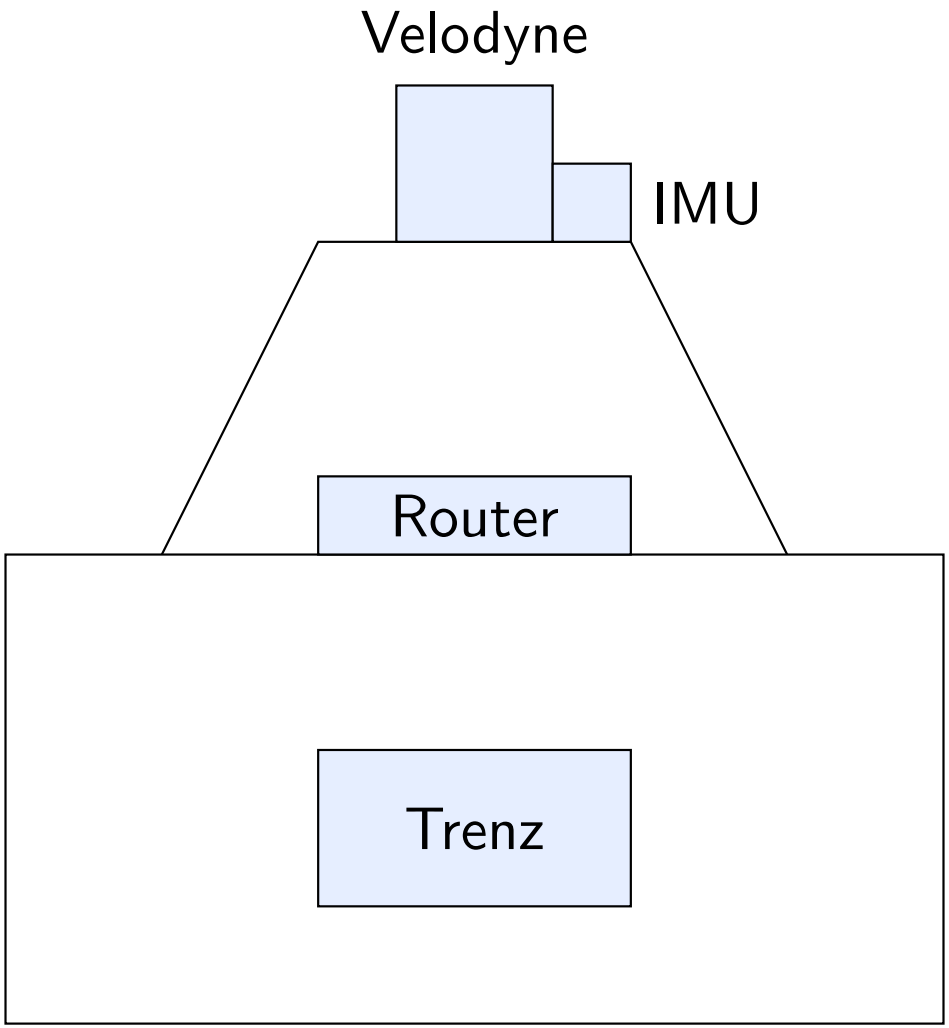
Registrierung

- Parallelisierbare Schritte der Schleife in HW (Berechnungen für alle Punkte)
- Matrixinvertierung in der Schleife bleibt in SW

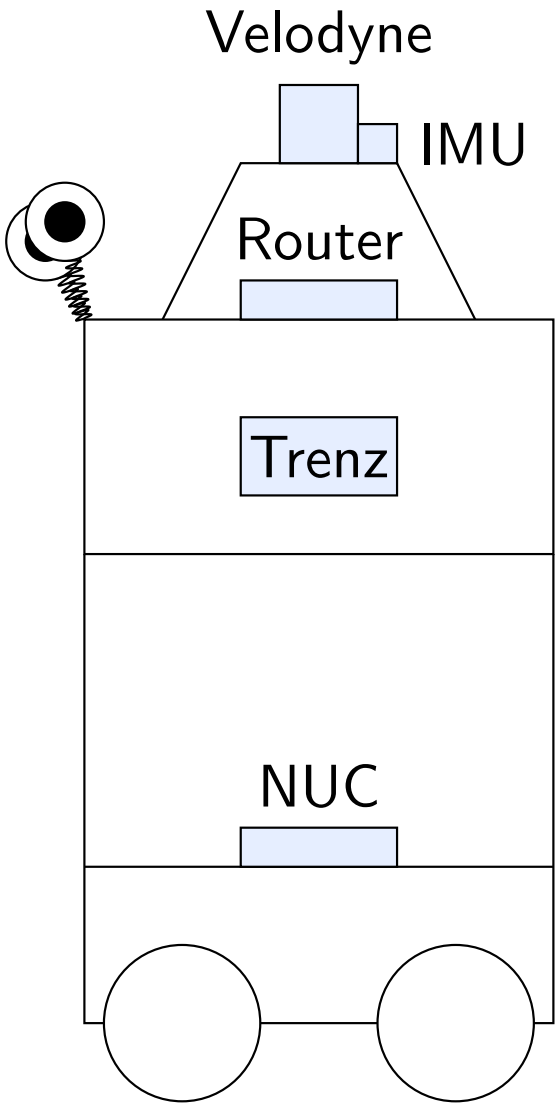
TSDF Update

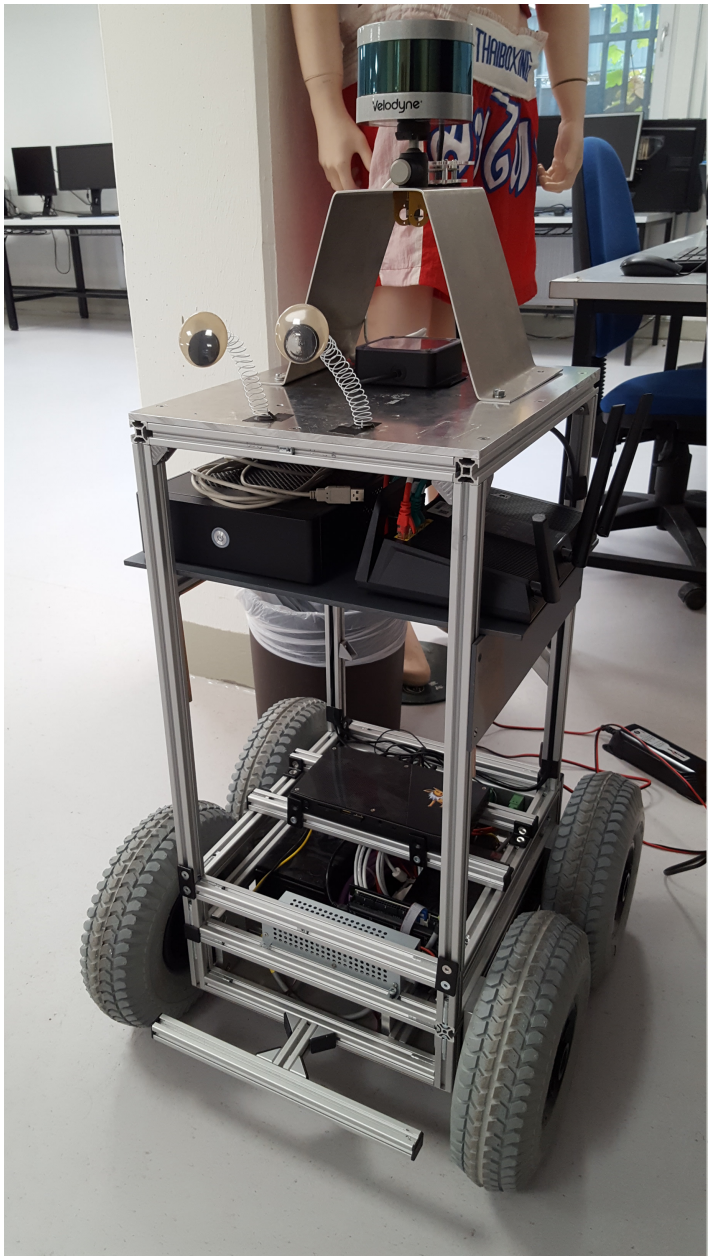
- Vollständig in HW
- Bresenham Algorithmus für die Iteration über Punkte entlang eines Strahls
- Synchronisation angepasst

FastSense Prototyp



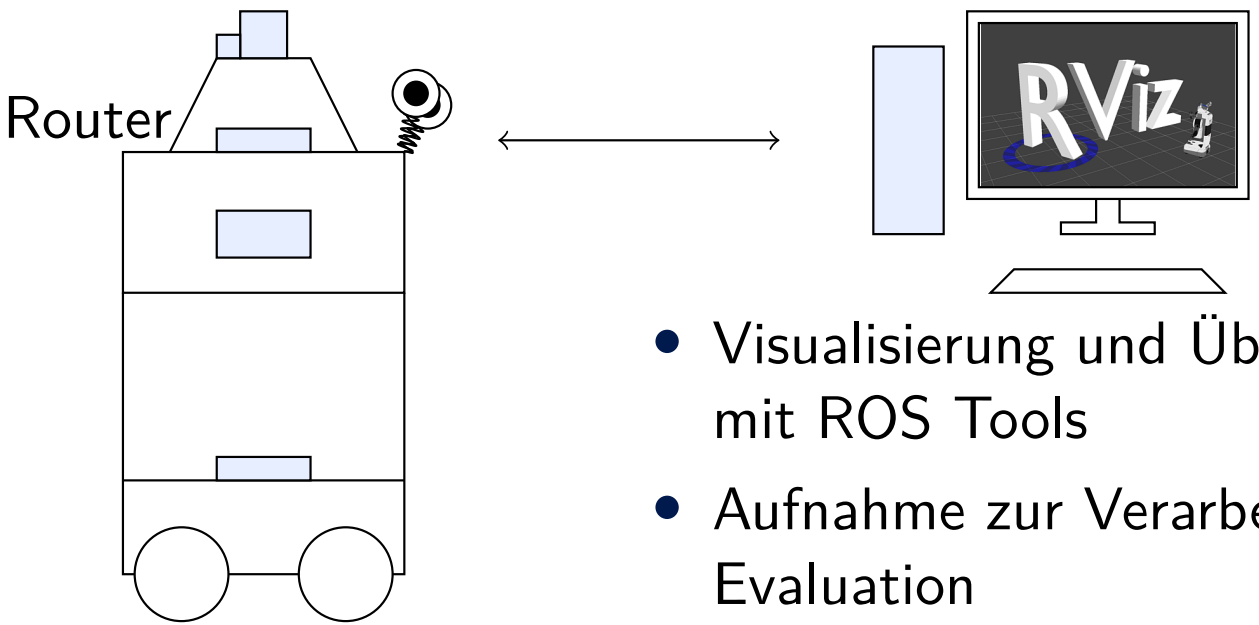
FastSense Prototyp





Kommunikation

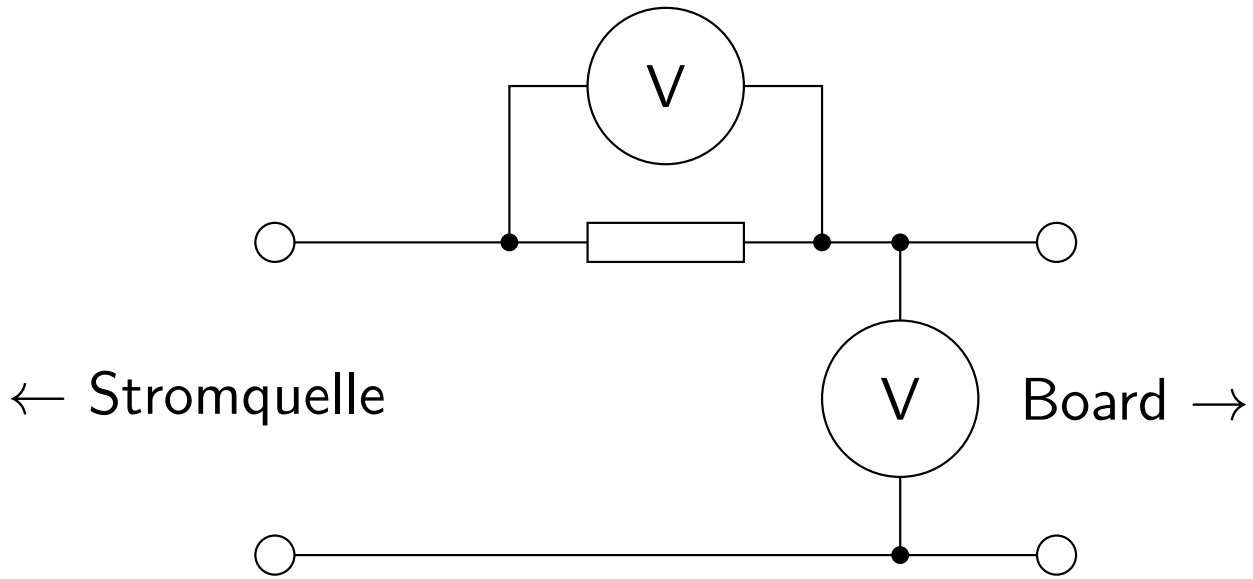
- Kommunikatioin via 'Bridge':
  Schnittstelle zwischen Sensordaten / berechneten Daten auf Trenz Board und einem Host Computer mit ROS
- Datenaustausch uber TCP, basierend auf ZeroMQ
  - LAN: Nahe real time TODO evaluieren
  - WLAN: TODO test mit ballermann WLAN empfaenger und ordentlichem Router



- Visualisierung und Überprüfung mit ROS Tools
- Aufnahme zur Verarbeitung und Evaluation

Evaluation

Strom



Strom

TODO: Ergebnisse

Zeit

Zeitmessung [ms]			
Abschnitt	Durchschnitt	Min	Max
Preprocessing	35	24	51
Registrierung	676	142	2894
Map Shift	105	0	1681
TSDF Update	337	319	346

Zeit

Vergleich Vitis – Realität [ms]		
Abschnitt	Vitis	Gemessen
Registrierung	0.905	3.667
TSDF Update	124.483	552.924

- Bei der Registrierung wird die meiste Zeit auf Speicher gewartet (95%)

Zeit

Vergleich verschiedener Plattformen [ms]			
Programm	FastSense	Prototyp	Prototyp
System	Board	NUC	Stand-PC
Preprocessing	35	???	???
Registrierung	676	???	???
Map Shift	105	???	???
TSDF Update	337	???	???

Fazit

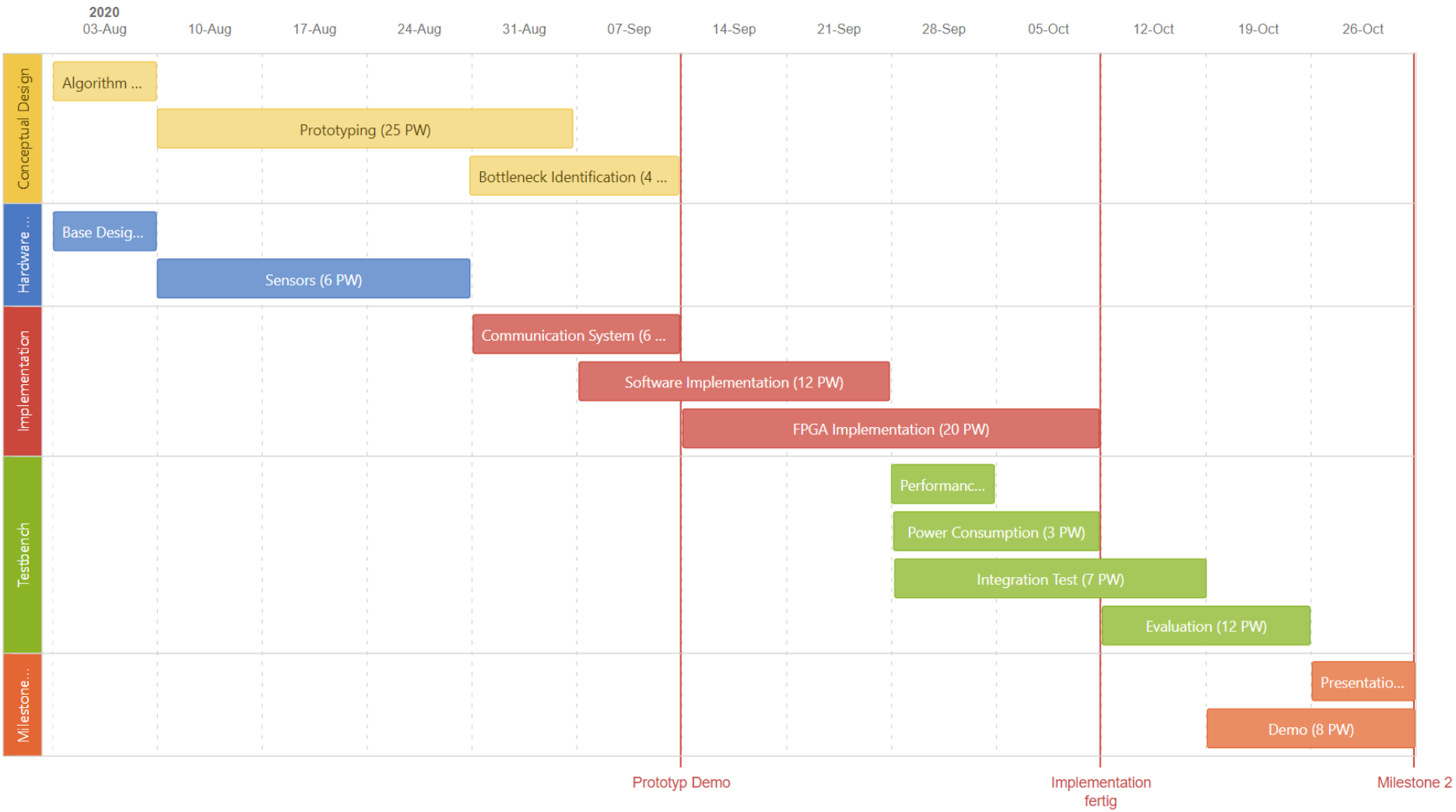
Bisherige Verbesserungen

- Registrierung
  - Auslagerung von Point to TSDF auf Hardware
  - Auslagerung von Pointcloud Transformation auf Hardware
- TSDF

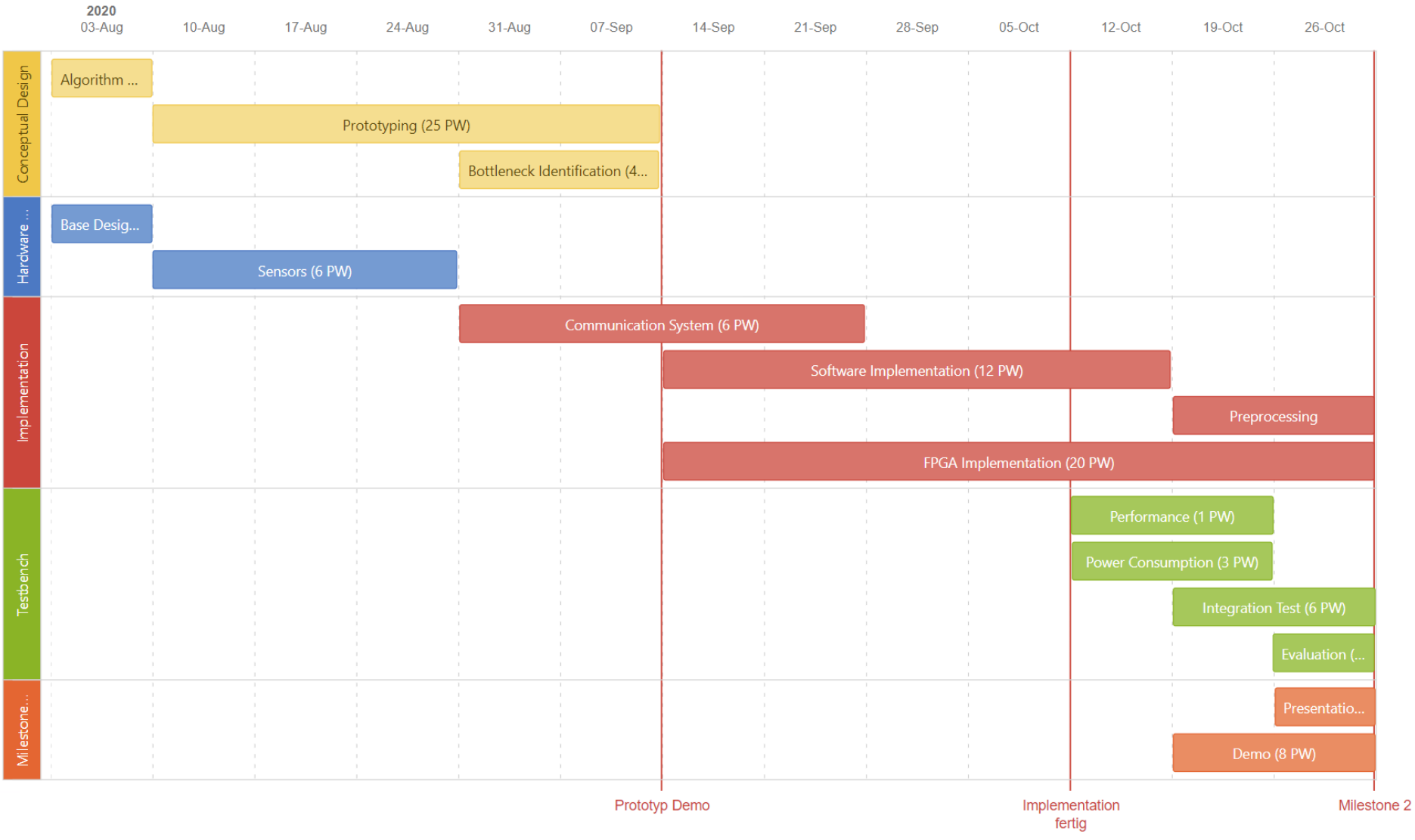
Verbesserungspotenzial

- Registrierung
  - Drift entfernen (aktuell noch leichter Drift (1cm/s) in alle 3 Richtungen)
  - Komplett in Hardware (Overhead ist fast dreimal so hoch wie der eigentliche Aufruf)

Projektmanagement



Projektmanagement



Ausblick / MS3

- Aufbau einer SLAM-Box mittels CAD
  - Nutzung als Sensor
  - Einfache Portierung zwischen Drohne, Roboter, Rucksack etc.
  - Festes Interface, einfache Bedienung, Kapselung
- Verbesserung und Optimierung des Algorithmus
- Mesh-Generierung auf Basis der TSDF Werte
- Loop Closing
- ???

TODO: Mehr Ideen für MS3

Quellen

[1] D. R. Canelhas, T. Stoyanov, and A. J. Lilienthal. “SDF Tracker: A parallel algorithm for on-line pose estimation and scene reconstruction from depth images”. In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2013, pp. 3671–3676. DOI: 10.1109/IROS.2013.6696880.