Softwarepraktikum SS2025
3D-Punktwolkenregistrierung mit Festkomma-Arithmetik

Tom Fleischmann, Jonas Wiesner und Yannik Winzer
(Betreuer: Prof. Dr. Andreas Nüchter)

JULIUS-MAXIMILIANS-UNIVERSITÄT WÜRZBURG

26.09.2025

Überblick

Ausgangssituation

Das 3DTK

- Das 3DTK arbeitet mit Punktwolken (Scans), wobei alle Punkte Koordinaten der Form (x, y, z) besitzen
- Die Scans stammen aus Sensoren; i.d.R. gibt es mehrere Scans
- Die verschiedenen Scans sollen konsistent zusammengesetzt werden. Dafür soll der Basis-ICP-Algorithmus (ohne Parallelisierung etc.) in Festkomma-Arithmetik implementiert werden

Durch die Verwendung der Festkomma-Arithmetik wird es möglich, den Algorithmus auf Platinenebene einzusetzen. Die Zielgruppe des Projekts ist der Informatik-Lehrstuhl 17 (Robotics) der Universität Würzburg.

Anforderungen an den Algorithmus

Iterative Closest Point (ICP)-Algorithmus zur Ausrichtung der Scans:

ICP (Iterative Closest Point)

Eingabe: zwei Scans M (Model) und D (Data)

1. bestimme für jeden Punkt von D den nächstgelegenen Punkt aus M (Punktkorrespondenzen)
2. rechne Fehlerfunktion mit Rotation und Translation aus

Die Schritte 1 und 2 werden iteriert, bis die Änderung des Fehlers kleiner als ein vorher festgelegtes ε ist oder die spezifizierte maximale Anzahl an Iterationen erreicht ist

Resultat: errechnete Rotation und Translation (wird für jeden Scan in der entsprechenden `.frames`-Datei gespeichert)

Aufgabenstellung

3D-Punktwolkenregistrierung mit Festkomma-Arithmetik

- Der ICP-Algorithmus soll als möglichst kleines, übersichtliches Programm in Festkomma-Arithmetik implementiert werden

Aufgabenstellung

3D-Punktwolkenregistrierung mit Festkomma-Arithmetik

- Der ICP-Algorithmus soll als möglichst kleines, übersichtliches Programm in Festkomma-Arithmetik implementiert werden
  - Festkomma-Arithmetik: *SystemC*-Library, d.h. feste Anzahl an Vor- und Nachkommastellen

Aufgabenstellung

3D-Punktwolkenregistrierung mit Festkomma-Arithmetik

- Der ICP-Algorithmus soll als möglichst kleines, übersichtliches Programm in Festkomma-Arithmetik implementiert werden
  - Festkomma-Arithmetik: *SystemC*-Library, d.h. feste Anzahl an Vor- und Nachkommastellen
  - Lösung des Minimalproblems (Rotation und Translation): *Approximations*-Methode

Aufgabenstellung

3D-Punktwolkenregistrierung mit Festkomma-Arithmetik

- Der ICP-Algorithmus soll als möglichst kleines, übersichtliches Programm in Festkomma-Arithmetik implementiert werden
  - Festkomma-Arithmetik: *SystemC*-Library, d.h. feste Anzahl an Vor- und Nachkommastellen
  - Lösung des Minimalproblems (Rotation und Translation): *Approximations*-Methode
  - Punktkorrespondenzen: *Brute Force*-Methode (doppelter for-loop)

Aufgabenstellung

3D-Punktwolkenregistrierung mit Festkomma-Arithmetik

- Der ICP-Algorithmus soll als möglichst kleines, übersichtliches Programm in Festkomma-Arithmetik implementiert werden
  - Festkomma-Arithmetik: *SystemC*-Library, d.h. feste Anzahl an Vor- und Nachkommastellen
  - Lösung des Minimalproblems (Rotation und Translation): *Approximations*-Methode
  - Punktkorrespondenzen: *Brute Force*-Methode (doppelter for-loop)
- Entwicklerhandbuch
- Hilfe-Anweisungen

Vorbereitungen

- Einbinden der SystemC-Library
  - Definieren des Festkomma Datentyps: `using f_float = sc_fixed<FIXED_WORD_LENGTH, FIXED_INT_WORD_LENGTH>;`

Vorbereitungen

- Einbinden der SystemC-Library
  - Definieren des Festkomma Datentyps: `using f_float = sc_fixed<FIXED_WORD_LENGTH, FIXED_INT_WORD_LENGTH>;`
- Erstellung der benötigten Dateien

Vorbereitungen

- Einbinden der SystemC-Library
  - Definieren des Festkomma Datentyps: `using f_float = sc_fixed<FIXED_WORD_LENGTH, FIXED_INT_WORD_LENGTH>;`
- Erstellung der benötigten Dateien
- Modifikation der CMake-Lists, sodass ...
  - ... `icpFixpoint` nach dem Build-Vorgang als Executable bereitgestellt wird
  - ... alle Abhängigkeiten miteingebunden werden
  - ... Einstellungsmöglichkeiten der Präzision `FIXED_INTEGER_WORD_LENGTH` und `FIXED_WORD_LENGTH` in cmake möglich sind
  - ... `HERON_ITERATIONS` zur Berechnung von Quadratwurzeln (mittels Heron-Verfahren) mit Festkomma-Arithmetik einstellbar ist

Struktur der neuen Dateien

- Ordner `src`: enthält die eigentlichen Code-Dateien
  - Ordner `sc_fixed`
    - `sc_fixed_converter.cc`: Konvertieren von `double`-Arrays in `f_float`-Arrays und -Vektoren
    - `sc_fixed_math.cc`: enthält mathematische Funktionen in SystemC-Festkomma-Arithmetik
    - `sc_ICP.cc`: `match`-Methode (ICP)
    - `sc_ICPapx.cc`: `Align`-Methode (richtet gegebenen Data-Scan am Source-Scan aus)
  - Ordner `slam6D`: Datei `icpFixpoint.cc`: Hauptdatei des Projekts für den ICP-Algorithmus
- Ordner `include/sc_fixed`: zugehörige Header-Dateien

Struktur der neuen Dateien

- Ordner `src`: enthält die eigentlichen Code-Dateien
  - Ordner `sc_fixed`
    - `sc_fixed_converter.cc`: Konvertieren von `double`-Arrays in `f_float`-Arrays und -Vektoren
    - `sc_fixed_math.cc`: enthält mathematische Funktionen in SystemC-Festkomma-Arithmetik
    - `sc_ICP.cc`: `match`-Methode (ICP)
    - `sc_ICPapx.cc`: `Align`-Methode (richtet gegebenen Data-Scan am Source-Scan aus)
  - Ordner `slam6D`: Datei `icpFixpoint.cc`: Hauptdatei des Projekts für den ICP-Algorithmus
- Ordner `include/sc_fixed`: zugehörige Header-Dateien
- Ordner `bin`: Executable `icpFixpoint`
- Ordner `doc`: Pflichtenheft und Entwicklerhandbuch

Aufruf und Ausgaben des Algorithmus

Ein möglicher Aufruf ist `bin/icpFixpoint -s 0 -e 2 dat` (mittels Kommandozeile), was beispielhaft zu folgendem Ergebnis in der zweiten `.frames`-Datei führt:

```
0.999756 -0.00854492 0.0078125 0 0.00830078 0.999756 0.0192871 0 -0.00830078 -0.0195312 0.999756 0 -7.99854 -15.353 337.973 1 0
0.999756 -0.00854492 0.0078125 0 0.00830078 0.999756 0.0192871 0 -0.00830078 -0.0195312 0.999756 0 -7.99854 -15.353 337.973 1 0
0.999756 -0.00854492 0.0078125 0 0.00830078 0.999756 0.0192871 0 -0.00830078 -0.0195312 0.999756 0 -7.99854 -15.353 337.973 1 0
0.999756 -0.00854492 0.0078125 0 0.00830078 0.999756 0.0192871 0 -0.00830078 -0.0195312 0.999756 0 -7.99854 -15.353 337.973 1 1
0.999756 -0.00854492 0.0078125 0 0.00830078 0.999512 0.0187988 0 -0.00830078 -0.0192871 0.999512 0 -7.11182 -15.2271 337.364 1 1
0.999756 -0.00854492 0.0078125 0 0.00830078 0.994629 0.0090332 0 -0.00830078 -0.0144043 0.994629 0 -6.50928 -16.4414 341.54 1 1
```

- Initialwerte werden für die Darstellung (`bin/show`) des vorherigen Scans kopiert
- Berechnung der Rotationsmatrix (Ergebnis in der letzten Zeile)
- Berechnung der Translationswerte (Ergebnis in der letzten Zeile)
- Zahl für die Darstellung mit `bin/show`

Erfüllung der Kriterien

3D-Punktwolkenregistrierung mit Festkomma-Arithmetik

- Der ICP-Algorithmus soll als möglichst kleines, übersichtliches Programm in Festkomma-Arithmetik implementiert werden ✓

Erfüllung der Kriterien

3D-Punktwolkenregistrierung mit Festkomma-Arithmetik

- Der ICP-Algorithmus soll als möglichst kleines, übersichtliches Programm in Festkomma-Arithmetik implementiert werden ✓
  - Festkomma-Arithmetik: *SystemC*-Library, d.h. feste Anzahl an Vor- und Nachkommastellen ✓

Erfüllung der Kriterien

3D-Punktwolkenregistrierung mit Festkomma-Arithmetik

- Der ICP-Algorithmus soll als möglichst kleines, übersichtliches Programm in Festkomma-Arithmetik implementiert werden ✓
  - Festkomma-Arithmetik: *SystemC*-Library, d.h. feste Anzahl an Vor- und Nachkommastellen ✓
  - Lösung des Minimalproblems (Rotation und Translation): *Approximations*-Methode ✓

Erfüllung der Kriterien

3D-Punktwolkenregistrierung mit Festkomma-Arithmetik

- Der ICP-Algorithmus soll als möglichst kleines, übersichtliches Programm in Festkomma-Arithmetik implementiert werden ✓
  - Festkomma-Arithmetik: *SystemC*-Library, d.h. feste Anzahl an Vor- und Nachkommastellen ✓
  - Lösung des Minimalproblems (Rotation und Translation): *Approximations*-Methode ✓
  - Punktkorrespondenzen: *Brute Force*-Methode (doppelter for-loop) ✓

Erfüllung der Kriterien

3D-Punktwolkenregistrierung mit Festkomma-Arithmetik

- Der ICP-Algorithmus soll als möglichst kleines, übersichtliches Programm in Festkomma-Arithmetik implementiert werden ✓
  - Festkomma-Arithmetik: *SystemC*-Library, d.h. feste Anzahl an Vor- und Nachkommastellen ✓
  - Lösung des Minimalproblems (Rotation und Translation): *Approximations*-Methode ✓
  - Punktkorrespondenzen: *Brute Force*-Methode (doppelter for-loop) ✓
- Entwicklerhandbuch ✓
- Hilfe-Anweisungen ✓

Ausblick

- Erweiterung des ICP-Algorithmus mit anderen Berechnungsmethoden (falls möglich)

Ausblick

- Erweiterung des ICP-Algorithmus mit anderen Berechnungsmethoden (falls möglich)
- Intensive Validierung des Algorithmus mit größeren Datensätzen

Gewonnene Einsichten

- Programmieren in C++ (verzeiht wenig)
- Verwenden von CMake (einarbeitungsintensiv)
- Kollaboratives Arbeiten mit Git(Hub) (praktisch)
- ... auf Linux

Gewonnene Einsichten

- Programmieren in C++ (verzeiht wenig)
- Verwenden von CMake (einarbeitungsintensiv)
- Kollaboratives Arbeiten mit Git(Hub) (praktisch)
- ... auf Linux
- Umgang mit Festkomma-Datentypen (Rundung, oft ungenau)

Gewonnene Einsichten

- Programmieren in C++ (verzeiht wenig)
- Verwenden von CMake (einarbeitungsintensiv)
- Kollaboratives Arbeiten mit Git(Hub) (praktisch)
- ... auf Linux
- Umgang mit Festkomma-Datentypen (Rundung, oft ungenau)
- Gemeinsames Arbeiten an einem Softwareprojekt (Teamarbeit)

Vielen Dank!

merci beaucoup!

Zeit für eure Fragen und tiefere Einblicke in den Code ...