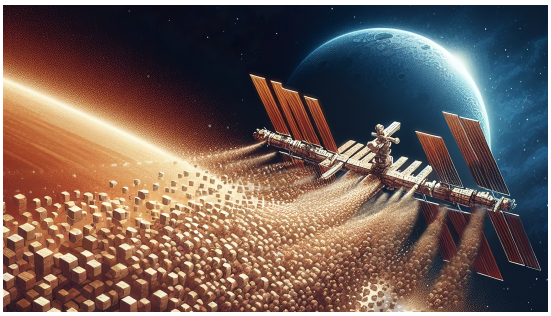
RIS - Riesige Infrastrukturen im Space

Aufgabenstellung Projektseminar Automatisierungstechnik WiSe 24/25

Version 2.0 (17. Oktober 2024) - WiMi: Linus Groß, M. Sc.

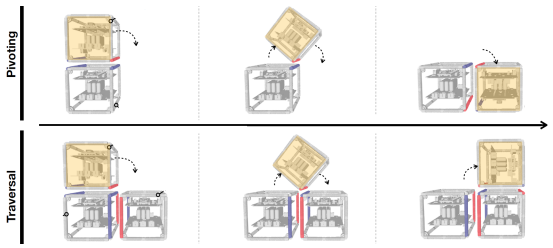
Einleitung

Willkommen zum Projekt RIS - Riesige Infrastrukturen im Space von der Forschungsgruppe r.i.s. (Research Institut for Space). Wir befinden uns im Jahr 2124 und in den letzten Jahrzehnten wurde hier Pionierarbeit zur selbstständigen Montage von Strukturen im Weltraum geleistet. Die Idee ist einfach, wie genial: Von der Erde werden leichte Komponenten ins All geschickt, welche sich dort selbstständig zu einer größeren Struktur zusammenbauen. Somit können riesige Bauprojekte wie Raumstationen, Raumschiffe oder Weltraumteleskope realisiert werden, um eine Infrastruktur im Weltall aufzubauen.



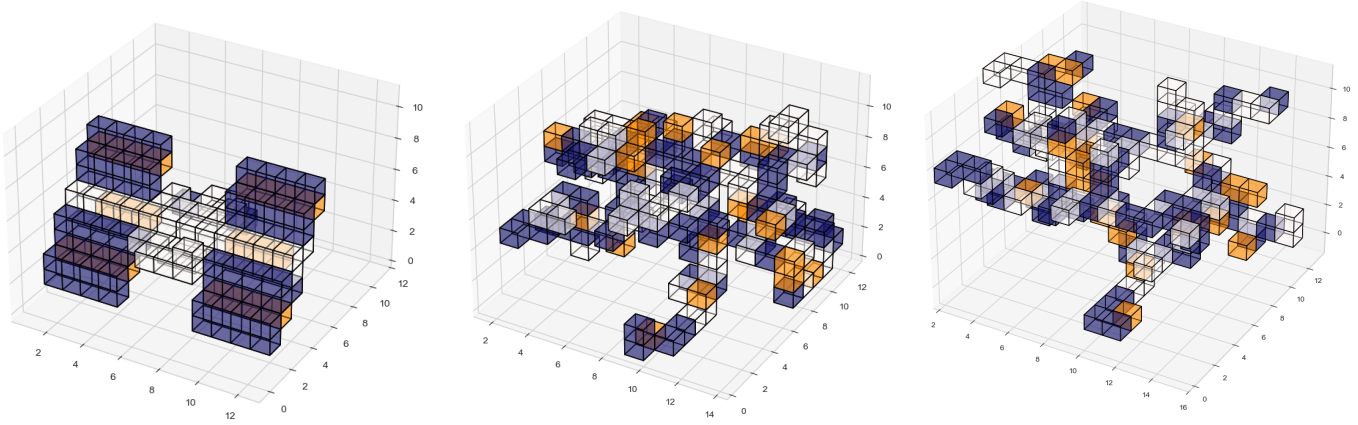
Die ISS wird aus Cubes zusammengebaut [1]

Jetzt rückt dieser Traum dank einer neuen Würfeltechnologie endlich näher: die sogenannten Cubes. Cubes sind kleine, programmierbare Würfel mit unterschiedlichen Designs und Materialien, die in der Lage sind, komplexe und voll funktionsfähige Strukturen zu bilden, wenn sie miteinander verbunden werden. Bereits im ferner Vergangenheit, im Jahre 2022, kam die Idee von programmierbaren Würfelkonfigurationen auf, wie folgendes Video zeigt. Die Cubes werden elektromagnetisch angesteuert, um sich modular zu verschiedenen Strukturen aufbauen zu können. Wie die rechte Abbildung zeigt, können die Cubes sich umeinander drehen (*pivoting*) oder entlang bereits vorhanden Cubes weiterbewegen (*traversal*).



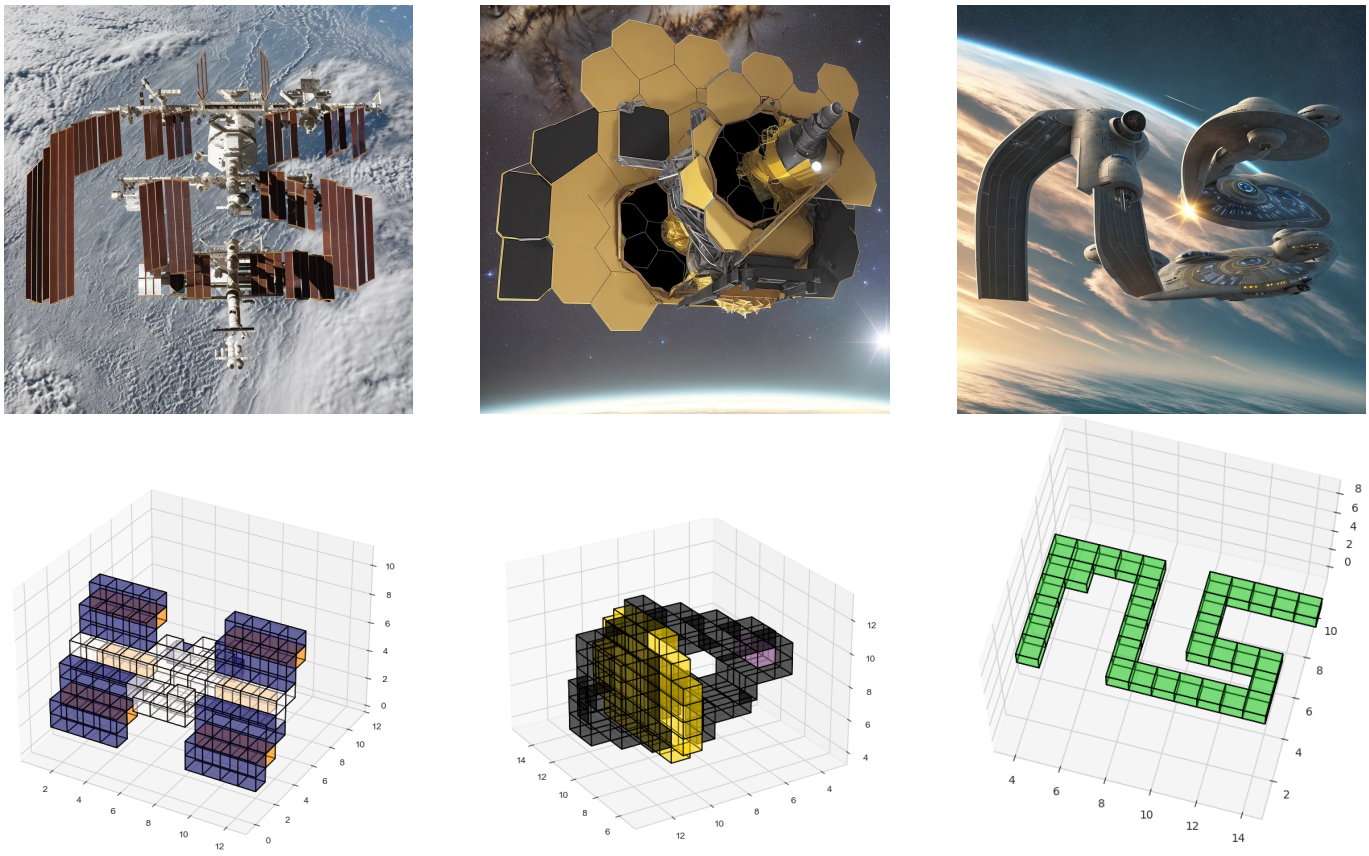
Bewegungsmöglichkeiten der Cubes [2]

Der fehlende Baustein für diese Technologie ist ein effizienter und skalierbarer Algorithmus zur Montage der Strukturen. Die Cubes werden als eine Ausgangsmenge in den Weltall hochgeschossen. Das Ziel ist, dass diese Ausgangsmenge an Cubes möglichst schnell und genau in eine vorgegeben Weltraumstruktur übergeht. Hierfür müssen die einzelnen Cubes programmiert werden: Welcher Cube dreht wann um welche Achse (*pivoting*) oder bewegt sich entlang anderer Cubes (*traversal*). Die Strukturen sind durch die Anzahl der Cubes, aus denen sie bestehen, die Arten von Cubes (die unterschiedliche Materialien oder Funktionen repräsentieren) und deren Position im 3D-Raum charakterisiert. Der von ihrer Forschungsgruppe zu entwerfende Algorithmus soll in der Lage sein, eine optimale Abfolge an Drehungen der Cubes zu finden, um die finale Struktur zu erlangen. [1]



(links) Zielkonstellation: Die ISS, welche aus den Cubes zusammengebaut werden soll. Verschiedene Arten von Cubes sind farblich unterschiedlich markiert.
(mitte) Ausgangskonstellation: In dieser Konfiguration starten die Cubes.
(rechts) Endkonstellation: Nach Ausführung einers (noch schlechten) Algorithmus ergibt sich diese Konstellation.

Als Beispielstrukturen wird die R-ISS, das neue James-Webb-Weltraumteleskop und das ris-Raumschiff vorgegeben. Diese Strukturen sollen mit Hilfe des zu entwickelnden Algorithmus optimal im Weltall aufgebaut werden. Dargestellt sind die Originalstrukturen, sowie die Zielzusammensetzung aus den einzelnen Cubes.



(links) Die Riesige International Space Station R-ISS, eine Neuauflage der bekannten ISS (148 Cubes mit 3 Cube-Typen).
(mitte) Die neue Version des James-Webb-Weltraumteleskops (JWST) (643 Cubes mit 6 Cube-Typen).
(rechts) Das neuartige ris-Raumschiff (34 Cubes mit 1 Cube-Typen).

Aufgabenstellung

Die Aufgabenstellung entsammt der ESA Space Optimisation Challenge (SpOC) 3: Programmable Cubes. Alle weiterführenden Informationen rund um die Aufgabenstellung befinden sich im obigen Link, bzw. im zugehörigen Github-Repository. Von dort kann die Aufgabenstellung und der Code heruntergeladen werden. (Wichtig: In diesem Projektseminar wird nur die Challenge 3 behandelt!)

Macht euch zunächst mit der IDE (z. B. PyCharm) vertraut und installiert alle notwendige Software und Packages. In der `Readme.md` ist die Aufgabenstellung im Detail beschrieben. Der komplette Code hierzu befindet sich in in `programmable_cubes_UDP.py` - die Aufgabenstellung, die Beschreibung der Cubes und die Simulation soweit schon programmiert und kommentiert. Macht euch hier mit den einzelnen Programmteilen vertraut. Hierfür ist es sinnvoll, die vier Tutorials in Jupyter-Notebook im Detail durchzuarbeiten.

Als Strukturen sind die Internationale Raumstation (ISS), das James-Webb-Weltraumteleskop (JWST) als auch die USS-Enterprise vorgegeben. Als zusätzliche Struktur geben wir noch das ris-Raumschiff vor. Das Ergebnis eurer Optimierung ist für jede Struktur eine Liste von Cube-Indizes und Kommandos (traversal oder pivoting), welche in einem Entscheidungsvektor x, auch Chromosom genannt, zusammengefasst wird. Hierbei ist die maximale Anzahl an Befehlen für jedes Problem beschränkt. Das Ziel ist es, mit möglich wenigen Befehlen die Zielkonfiguration so gut wie möglich abzudecken. Für eine qualitative Bewertung wird eine Fitnessfunktion $f(x)$ herangezogen, welche abhängig vom Entscheidungsvektor x einen Score liefert. Der theoretisch bestmögliche Score ist

$$f(x_{\text{opt}}) = -1,$$

der Default-Score mit der Ausgangskonfiguration ist

$$f(x_0) = 0.$$

Für die Evaluation eurer Ergebnisse verwenden wir dieselbe Fitnessfunktion.

Der Wettbewerb besteht aus 3 Teilaufgaben:

- **Aufgabe 1 - Optimierte Transformation von 4 Strukturen**: In den 3 vorgegebenen Strukturen (ISS, JWST, Enterprise), sowie dem ris-Raumschiff den besten Score erzielen. Die Lösung wird hierbei auf der ESA-Website hochgeladen. Jedes Team wählt einen *Teamnamen* und lädt die Ergebnisse unter dem Namen **TUDa_Space_*Teamname*** auf der ESA-Website hoch.
- **Aufgabe 2 - Echtzeitanwendung:** Nach Abgabe des Projekts wird eine neue, bisher unbekannte Struktur vorgestellt. Euer Programm[1] hat nun genau 20 Minuten Zeit, eine bestmögliche Lösung zu finden. Nach 20 Minuten brechen wir euren Algorithmus ab (*anytime algorithm*). Die dann verfügbare Lösung (Entscheidungsvektor x) wird verwendet.
- **Aufgabe 3 - Visualisierung:** Es soll eine GUI erstellt werden, mit der die Lösung für eine Struktur visualisiert wird. Die GUI soll hierbei als Teil der Abschlusspräsentation vorgestellt werden (ca. 5 min). Hierbei könnt ihr eurer Kreativität freien Lauf lassen, was genau mit der GUI angezeigt wird und was ihr visualisieren wollt - es gibt keine Vorgabe für diese Aufgabe. Als Startpunkt könnt ihr Tutorial 1 nehmen, wo aus Python-Plots ein GIF erstellt wird. Einige paar Ideen von unserer Seite, um eure Kreativität anzuregen:
  - Visualisierung bzw. Animation des zeitlichen Aufbaus der Struktur aus den Cubes.
  - Zuweisung von verschiedenen Texturen zu den Cube-Typen.
  - schwenkbare und zoombare 3D-Ansicht im Weltraum.

Code

- Die Aufgabenstellung ist in `Python 3.8.8` programmiert. Es steht euch frei, eure Optimierung auch in anderen Programmiersprachen durchzuführen.
- Der Code für die Aufgabenstellung kann über das offizielle Git-Repository heruntergeladen/geforked/gecloned werden.
- Für eure Codeverwaltung bietet sich ein eigenes Git-Repository an, hierfür kann sich auf GitLab mit der TU-ID eingeloggt werden.
- Eine gute IDE für die Programmierung von Python ist PyCharm. Auch hier bekommt man eine kostenlose Professional-Version mit der TU-Mailadresse - eine Anleitung findet sich hier. Achtet darauf, die Professional-Version zu installieren - dies ist notwendig, um beispielsweise die Tutorials mit Jupyter-Notebook in PyCharm auszuführen. Anbei ist auch eine Anleitung zur Nutzung von Jupyter-Notebooks in PyCharm.
- Eine gute GUI für Git ist SmartGit. Mit der TU-Mailadresse kann man sich konstenlos eine Non-Commercial License besorgen, indem man sich hier unter `Educational Institution` mit der TU-Mailadresse registriert. Man erhält eine Lizenzdatei zugeschickt, welche man anschließend in SmartGit hinterlegen kann.
- Falls ihr noch nie mit Git gearbeitet habt ist LearnGitBranching eine gute Website, um spielend die Grundlagen dafür zu erlernen.
- Der Code basiert auf `Python 3.8.8` und folgenden Libraries:

```
numba >= 0.56.4
numpy >= 1.23.5
json >= 2.0.9
matplotlib >= 3.6.3
```

- Außerdem wird Pygmo benötigt. Für die Installation bietet es sich an, mit Conda eine passende Environment zu erstellen.
- Die `.json`- und `.npy`-Dateien für das ris-Raumschiff können in Moodle heruntergeladen werden und müssen entsprechend in die Ordnerstruktur eingegliedert werden:

```
./data/RIS/
    Initial_Config.npy
    Initial_Cube_Types.npy
    Target_Config.npy
    Target_Cube_Types.npy
./data/problems/RIS.json
```

[1] Es kann dasselbe sein, dass ihr in Aufgabe 1 verwendet habt, es kann aber auch ein anderes sein.

Organisatorisches

- Alle weiteren Details zur Aufgabenstellung werden bei der Auftaktveranstaltung am

  Donnerstag, den 17.10.2024 16.15-17.55 Uhr, Raum S204|213

  bekanntgegeben. Bitte beachtet die restlichen organisatorischen Hinweise, insbesondere die Fristen zum An- und Abmelden zum Projektseminar auf der Website zum Projektseminar.
- Die Lehrveranstaltung ist als Wettbewerb organisiert, in dem kleine Projektgruppen jeweils das gleiche Thema bearbeiten. Die erarbeiteten Lösungen werden hierbei im Rahmen der Abschlusspräsentationen miteinander verglichen.
- Es gibt keine individuelle fachliche Betreuung für die einzelnen Gruppen! Die Lösungsansätze und das hierfür benötigte Basiswissen sollen und müssen eigenständig von der Gruppe erarbeitet werden.
- Organisatorische Fragen und Fragen rund um die Aufgabenstellung werden natürlich beantwortet.
- Der Bearbeitungszeitraum erstreckt sich vom 17.10.2024 bis zum 03.02.2025. Eine erfolgreiche Projektarbeit ist Voraussetzung für die Bewertung des Projektseminars.
- Pro Gruppe ist eine 10 seitige Ausarbeitung anzufertigen. Eine Vorlage dafür wird per Moodle bereitgestellt. Die Ausarbeitungen müssen bis zum 03.02.2025, 12:00 MEZ in Moodle hochgeladen werden. Jede Gruppe gibt eine Ausarbeitung ab. Nähere Angaben zum Inhalt und der Struktur der Ausarbeitung werden während der Projektarbeit gegeben.
- Im Rahmen der Abschlussveranstaltung hält jede Gruppe zudem eine 20-minütige Präsentation.
- Es gibt keine individuelle fachliche Betreuung für die einzelnen Gruppen. Die Lösungsansätze müssen von jeder Gruppe selbstständig erarbeitet werden. Organisatorische Fragen werden natürlich beantwortet.
- Im Lernzentrum und PC-Pool des Institus (4. Stock in S3|10) werden Arbeitsplätze angeboten. Bei Bedarf kann auch zusätzliche Hardware (Rechner) zur Verfügung gestellt werden.

Bitte beachtet, dass wir uns kleinere Änderungen der Aufgabenstellung vorbehalten. Schaut, dass ihr stets die aktuelle Version der Aufgabenstellung verwendet. Ansprechpartner:

- Linus Groß, M.Sc. - Kontaktseite
- linus.gross@tu-darmstadt.de
- Organisatorische Informationen während des Semesters: Moodle-Forum

Patch der Mission