Autonomous Object Inspection with Mobile Robots for 3D Reconstruction and Image Data Acquisition

Bachelor Thesis

Ian Schmetkamp [1]
Advisor: Philip Keller [2]
Supervisor: Prof. Dr.-Ing. Rüdiger Dillmann [2]

Karlsruhe Institute of Technology

FZI Research Center for Information Technology

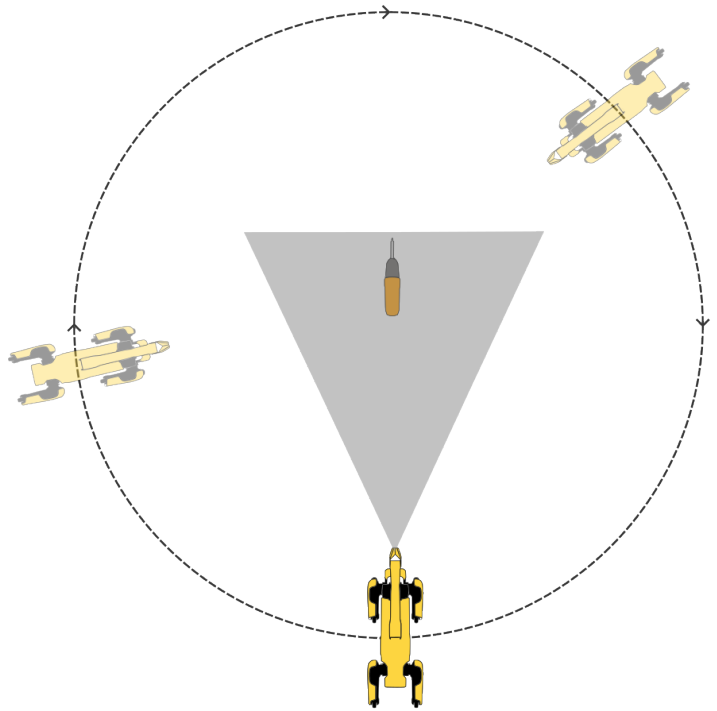
24. Januar 2025

Inhalt

Aufgabe

Ziel

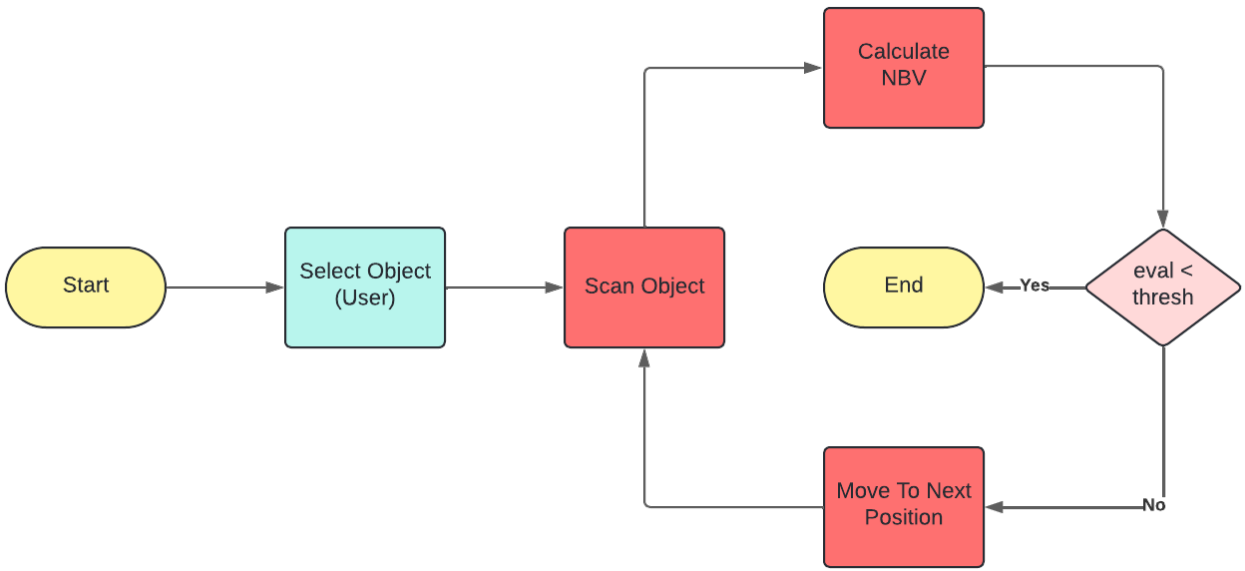
- autonom Bilder von Object aufnehmen
- Bilder aus verschiedenen Perspektiven
- **Roboter (Spot, Turtlebot, etc.) muss nächste, vorteilhafte Perspektive berechnen**



Ansatz

Ansatz

- Mögliche Positionen evaluieren
- Abschätzen wie viel Informationen in möglicher Position gesehen wird (Stichwort: Next-Best-View)
- Andere Faktoren für Positionen evaluieren (Distanz, Überlappung, Sichtbarkeit des Bekannten etc.)
- Zur besten Position gehen



NBV: Object Reconstruction

Klassische Ansätze

- Roboter mit Tiefensensor
- Generierung von Kandidaten Position, meist auf Kugel um Objektzentrum
- Testen der Kandidaten auf utility function
- Zum besten Kandidaten bewegen
- Nutzen meist Voxel-Karte (Octomap) und Bounding Box um Objekt

[4]

Machine-Learning Ansätze

- Nächste Sensorpose durch Punktwolke oder Voxel-Karte vorhersagen
- PC-NBV [3], NBV-Net [1]
- Bounding Box

NBV: Object Reconstruction

Unterschiede zu meinem Ansatz

- keine Bounding Box/ keine Information über Größe des Objekts
- VDB-Mapping

Herausforderungen

- besetzte Voxel entweder Teil des Objekts oder Teil der Umgebung
- mehr Information zu überprüfen $\rightarrow$ höhere Laufzeit
- Generierung von Kandidaten auf Kugel nicht möglich, da kein Objektzentrum und keine Objektdimension bekannt

Algorithmen

Algorithm NBV

Require: pixel: Tuple<int>, thresh

```
ray = convertPixelToRay(pixel)
scan()
origin = raytrace(camera.position, ray)
while eval > thresh do
    scan()
    surfaceVoxel, frontierVoxel = breadthFirstSearch(origin)
    frontiers = groupFrontiers(frontierVoxel)
    candidates = generateCandidates(frontiers)
    eval, c = max(evaluateCandidates(candidates, surfaceVoxel))
    moveTo(c)
end while
```

Definitionen

Visible unknown Voxel

- unbekannte Voxel, die von einem anderen Punkt als erste gesehen werden
- unbekannte Voxel benachbart zu einem freien Voxel

[2]

Frontier Voxel

- visible unknown Voxel, die benachbart zu besetzen Voxel sind
- benachbarte Frontier Voxel bilden eine Frontier

[2]

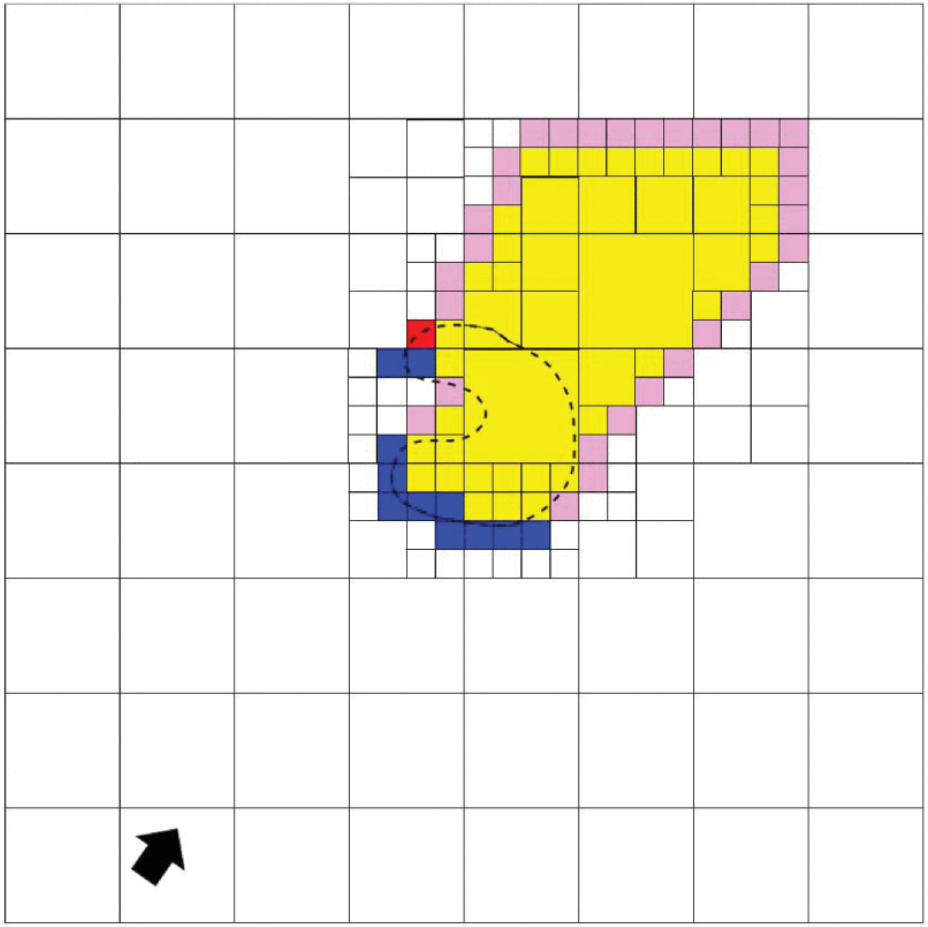


Abbildung: occupied voxel blue, unknown voxel yellow, **visible unknown** voxel pink, **frontier** voxel red [2]

Algorithmen

Frontier Voxel

- unbekannte Voxel, die benachbart zu freien **und** besetzen Voxel sind
- benachbarte Frontier Voxel bilden eine Frontier

1. Breitensuche

- Finde alle besetzen Voxel, die über besetze Voxel mit Origin verbunden sind
- Findet die bisher bekannte Oberfläche
- Filtert den Boden heraus
- Markiert alle gefundenen unbekannte Voxel die freien Nachbarn haben

Algorithmen

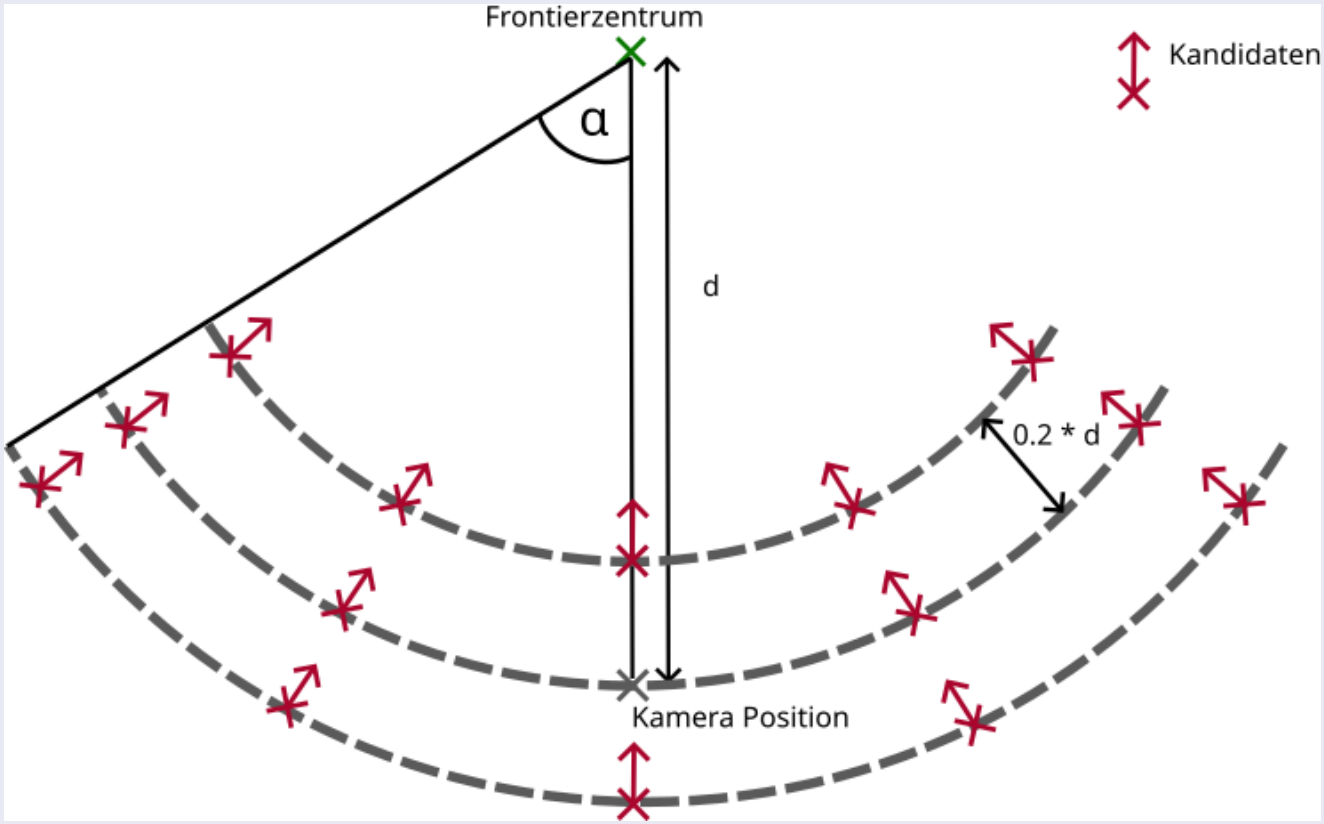
Frontier Voxel

- unbekannte Voxel, die benachbart zu freien **und** besetzen Voxel sind
- benachbarte Frontier Voxel bilden eine Frontier

2. Breitensuche: Frontier Voxel gruppieren

- 2 verschachtelte Breitensuchen
- gruppiert alle benachbarten Frontier Voxel zu einer Frontier
- Berechnet Zentrum der Frontier

Kandidaten Positionen generieren



- Sampling zwischen fixen Distanzen, abhängig von aktueller Distanz d
- Sampling zwischen maximalem Ausschlagswinkel α

Algorithmen

Algorithm Evaluate Candidates Part 1

Require: candidates, surfaceVoxel

```
    for c ∈ candidates do
2:      num_surface = 0
        for s ∈ surfaceVoxel do
4:          if isPartOfViewFrustum(s, c) then
                num_surface++
6:          end if
        end for
8:  end for
```

Algorithmen

Algorithm Evaluate Candidates Part 2

Require: candidates, surfaceVoxel
    num_seen_surface, num_seen_unknown = 0
2: **for** c ∈ candidates **do**
        for pixel ∈ Sensor **do**
4:          ray = rotateToPose(convertPixelToRay(pixel), c)
            point = raytraceInRange(c, ray)
6:          **if** point ∈ surfaceVoxel **then**
                num_seen_surface++
8:          **end if**
            if point is unknown **then**
10:             num_seen_unknown++
            end if
12:     **end for**
    end for

Evaluation

Für ein Kandidaten v:

$$f(v) = \frac{1}{1 + \mathrm{d(v)}} \cdot \frac{n_u(v)}{1 + \max_{v'}(n_u(v'))} \cdot \left(\frac{n_s(v)}{\sum_{v'} n_s(v')}\right)^2 \cdot r(v) \cdot p(v)$$

Mit

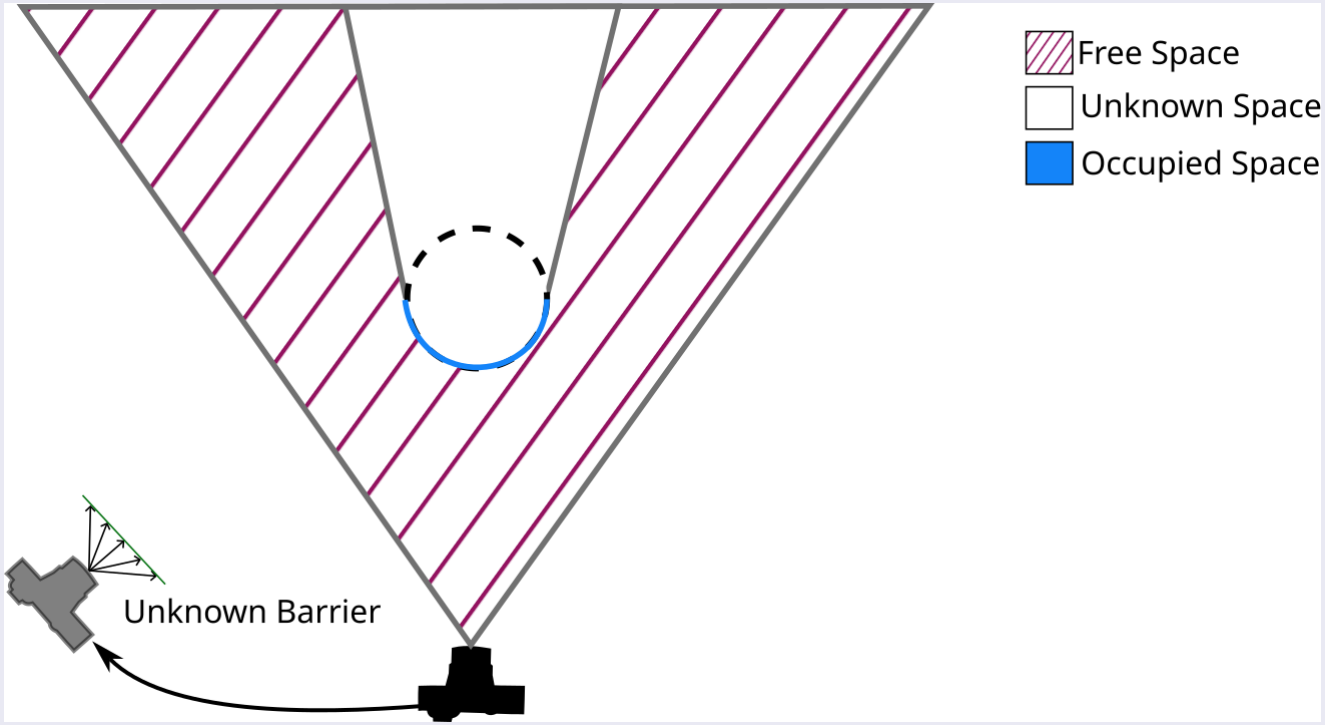
- n_u Anzahl getroffener unbekannter Voxel,
- $n_{r,s}$ Anzahl getroffener surfaceVoxel,
- n_s Anzahl surfaceVoxel im Sichtkörper,

$$p(v) = \begin{cases} 1 & \text{Wenn } v \text{ erreichbar ist} \\ 0 & \text{sonst} \end{cases}$$

und

$$r(v) = \begin{cases} 1 & \text{Wenn } n_{r,s}(v) > \sum_{v'} n_{r,s}(v') \cdot 0.1 \\ 0 & \text{sonst} \end{cases}$$

Problem: Unknown Barrier



- Alle Strahlen von neuer Kamera Position treffen auf unbekannte Voxel
- Interessanter Bereich hinter Object
- Je länger das Programm läuft desto komplizierter wird Körper hinter Objekt

[1] Miguel Mendoza u. a. „Supervised learning of the next-best-view for 3d object reconstruction“. In: *Pattern Recognition Letters* 133 (Mai 2020), S. 224–231. ISSN: 0167-8655. DOI: 10.1016/j.patrec.2020.02.024. URL: https://www.sciencedirect.com/science/article/pii/S0167865518305531 (besucht am 18. 09. 2024).

[2] J. Irving Vasquez-Gomez. „VPL: A View Planning Library for Automated 3D Reconstruction“. In: Nov. 2020. URL: https://ieeexplore.ieee.org/document/9359430/?arnumber=9359430 (besucht am 08. 10. 2024).

[3] Rui Zeng, Wang Zhao und Yong-Jin Liu. „PC-NBV: A Point Cloud Based Deep Network for Efficient Next Best View Planning“. In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. ISSN: 2153-0866. Okt. 2020, S. 7050–7057. DOI: 10.1109/IROS45743.2020.9340916. URL: https://ieeexplore.ieee.org/document/9340916/?arnumber=9340916 (besucht am 08. 10. 2024).

[4] Rui Zeng u. a. „View planning in robot active vision: A survey of systems, algorithms, and applications“. en. In: *Computational Visual Media* 6.3 (Sep. 2020), S. 225–245. ISSN: 2096-0662. DOI: 10.1007/s41095-020-0179-3. URL: https://doi.org/10.1007/s41095-020-0179-3 (besucht am 17. 09. 2024).