

## Aufgabenstellung und Ausgangspunkt

Dieses Projekt wurde als Weiterentwicklung einer Bachelorarbeit geplant. Der Ausgangspunkt war eine Kinect Applikation fürs 3D-Scannen von Objekten mit einem Pfeil-Overlay als Hilfe, der in Richtung des berechneten Next-Best-Views zeigte. Die ursprüngliche Vision war es, eine Crossplattform-App mit Flutter zu entwickeln. Flutter bietet die Möglichkeit, mit nur einem Code-Basis sowohl iOS als auch Android abzudecken. Allerdings zeigte sich schnell, dass die Einbindung nativer C++-Bibliotheken – insbesondere einer so komplexen wie RTAB-Map – in Flutter sehr schwierig ist. Zwar können Flutter-Apps über „Platform Channels“ auf nativen Code zugreifen, aber die Integration eines großen C++-Kerns mit OpenCV, PCL und Boost-Abhängigkeiten ist mobil kaum realistisch.

Zudem war ursprünglich geplant, für die NBV-Berechnung ein bereits vorhandenes Python-Paket zu nutzen, das im Kinect-Prototyp verwendet wurde. Auch hier gab es auf mobilen Plattformen Probleme: Python ist in Android und iOS nicht nativ eingebunden, die Interpreter-Integration ist aufwendig, speicherintensiv und würde den App-Store-Richtlinien widersprechen. Deshalb wurde entschieden, die NBV-Logik direkt in C++ neu zu implementieren, innerhalb des RTAB-Map-Kerns. Dadurch konnten wir die bestehende native Infrastruktur nutzen und auf mobilen Geräten effizient arbeiten. Der Nachteil dieser Herangehensweise, dass zwei eigenständige Applikationen funktionstüchtig gebracht werden mussten, wurde deshalb in Kauf genommen. RTABMap kam in diesem Moment sehr gelegen, da es bereits zwei, teilweise recht veraltete und nicht mehr wirklich gewartete aber dennoch sehr hilfreiche, separate Applikationen mitbringt. Eine Swift Applikation und eine in einer etwas veralteten Java Variante, die noch ohne Gradle und deshalb mit Cmake arbeitet.

## Herangehensweise und Challenges

Von Anfang an gab es Integrationsschwierigkeiten. Es musste aufgrund der tiefgreifenden Sensordaten, die für den NBV-Algorithmus notwendig waren, direkt in C++ Methoden implementiert werden, auf die die nativen Applikationen dann zugreifen konnten. Somit mussten in beiden Applikationen Brücken geschrieben werden, die diese Zugriffe erlauben. Hierbei war es essentiell, ohne jedliche code completion oder IDE Warnings die richtigen Formate und Syntax zu finden.

Teilweise wurden eigenartige Datentypen oder auch Formate vorausgesetzt, über die man zuerst Recherchieren musste (bspw. Jarrays oder dass man `const float*` nicht als `float*` übergeben werden können, es aber keine anständige Fehlermeldung dafür gibt) Bei der IOS app wurde ObjectiveC++ verwendet und für Android war es eine .jini, die dies übernahm. Das Debugging in IOS war mit Xcode noch recht angenehm, da es Logs und Crash-Informationen in der IDE gab, für die Android app ohne wirkliche Android Studio Integration war es jedoch immer notwendig die LogCat Files zu analysieren, um Absturz Gründe nachvollziehen zu können, die sonst einfach nirgends aufscheinen würden (beispielsweise Javas spezielles Handling von `Float.NaN` Werten das manchmal zu Problemen führte). Auch die Render Frameworks waren sehr unterschiedlich, während die IOS Applikation mit WebGL arbeitete, benutzte die Android App Tango, was wieder zusätzliche Recherchen mitsichzog.

## NextBestView Algorithmus

Die Funktion `getNextBestViewPoseOnly()`, die in C++ für beide Applikationen erstellt wurde, implementiert eine minimale Strategie für „Next Best View“ (NBV) unter Verwendung der Occupancy-Grids von RTAB-Map. Sie fragt den Graphen mit `withGrid=true` ab, um Signaturen zu erhalten, die lokale 2D-Occupancy-Grids enthalten (Boden, Hindernisse, freie Zellen, unbekannte Zellen). Falls nötig, werden diese Grids entkomprimiert, um direkten Zugriff auf die Zellwerte zu haben.

Der Algorithmus durchsucht anschließend das Grid nach sogenannten „Frontier-Zellen“. Das sind unbekannte Zellen, die an mindestens eine freie Zelle angrenzen. Solche Zellen markieren die Grenze zwischen bereits kartierten und noch unbekannten Bereichen – also Regionen, in denen neue Informationen gewonnen werden können. Zur Vereinfachung bewertet die Funktion jede gefundene Frontier-Zelle ausschließlich anhand der negativen euklidischen Distanz zur aktuellen Pose (nähere Zellen erhalten eine höhere Bewertung). Die nächstgelegene Frontier-Zelle wird dann als „Next Best View“ ausgewählt und ihre Koordinaten als Zielposition zurückgegeben. Wenn keine Frontier gefunden wird, gibt die Funktion stattdessen die aktuelle Pose zurück.

Zudem wurde eine Fallback-Logik implementiert, die es erlaubt auch Fälle in denen Voxel Berechnungen fehlschlagen, oder keine Tiefeninformation berechnet werden kann (kein Lidar oder StereoVision) ebenfalls handzuhaben. Dieser Fallback Algorithmus nutzt eine vereinfachte Herangehensweise, in der er sich nur auf die visuellen Merkmale (Features/Wörter) aus den RGB-Bildern stützt. Ist es also dem NBV-Algorithmus nicht möglich, freie, belegte und unbekannte Zellen im Raum zu unterscheiden und daraus sinnvolle „Frontier“-Positionen abzuleiten so schließt der RGB-Fallback diese Lücke, indem er Bildmerkmale als Näherung verwendet – weniger präzise, aber immer noch nützlich für Exploration oder Navigation.

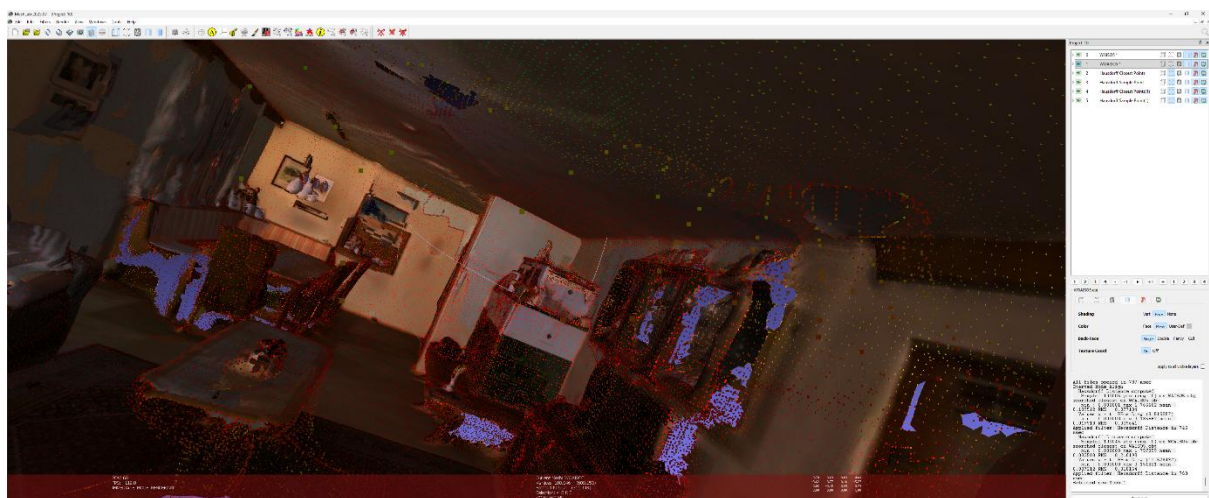
## User Study

Zur Testung der NBV Hilfe, wurde eine User Study durchgeführt mit 5 Leuten. 3 davon sind in einem schwer informatiklastigem Beruf, 2 davon haben fast keine beruflichen Berührungspunkte mit Applikationen. Es wurden jeweils verschiedene Objekte getestet und immer derselbe Raum im Inside-Out Testing. Somit sind pro Person 8 Scans angefallen, jeweils ein Scan mit Pfeil und einer ohne für jeweils Objekt-Scanning und Inside-Out Scanning mit jeweils dem Android Gerät und dem Ipad. Alle Scans wurden zeitlich mit 1 Minute begrenzt, diese Entscheidung wurde getroffen, damit nicht zu viele Redundanzen entstehen, die die Scanqualitäten zu stark verbessern. Dadurch sollte klarer ersichtlich werden, wie viel es wirklich hilft, einen mutmaßlich effizienteren Weg mit NBV zu wählen. Als Auswertungskriterium wurde die Hausdorff-Distanz, berechnet von Meshlab, nachdem die zwei Scans zueinander aligned wurden, gewählt. Sie liefert in mean, max, min Werten die Abweichungen zwischen den jeweiligen Scans mit und ohne NBV in jeweils eine Richtung und es kann durch die richtungsunterschiede festgemacht werden, welcher Scan, mehr Abdeckung und Präzision besitzt.

## IOS App Test (Mit Lidar)

Die IOS Scans waren alle samt Qualitativ sehr hochwertig und nahezu vollständig, auch beim Inside Out des, doch nicht ganz so kleinen Raumes.

## Ergebnisse -> Vergleich mit NBV Arrow und ohne NBV Arrow



Beispiel: Raumsan von Person 3 IOS mit Hausdorff Distance eingeblendet.

Legende: WA.. -> With Arrow / WOA.. -> WithOut Arrow

Kategorie	Richtung	Mittel [m]	RMS [m]	Max [m]	BBox Diag [m]	Mittel (norm.)	RMS (norm.)	Interpretation
Objekt 1	WA1 -> WOA1	0.122	0.24	2.73	8.71	0.0014	0.0277	Sehr gute Übereinstimmung; NBV-Scan leicht bessere Abdeckung
	WOA1 -> WA1	0.080	0.201	2.82	9.601	0.0083	0.0216	
Inside- Out 1	WAIO1 → WOAISO1	0.066	0.102	0.73	11.33	0.0058	0.0090	Sehr gute Übereinstimmung; NBV-Scan weicht weniger als 1 % vom Baseline-Modell ab.
	WOAISO1 → WAIO1	0.084	0.155	1.15	9.80	0.0086	0.0158	Baseline wird weitgehend vom NBV- Scan abgedeckt, NBV erfasst etwas mehr Fläche.
Objekt 2	WA2 → WOA2	0.135	0.257	3.64	10.11	0.0134	0.0254	Gute geometrische Übereinstimmung; NBV ergänzt zusätzliche Details.
	WOA2 → WA2	0.164	0.291	3.70	9.72	0.0168	0.0299	Etwas höhere Abweichung in Gegenrichtung; Abdeckung durch NBV bleibt vollständig.
Inside- Out 2	WAISO2 → WOAISO3	0.220	0.291	1.56	11.88	0.0186	0.0245	Moderate Abweichung; NBV erfasst größere Bereiche, jedoch

								geringfügig unpräziser.
	WOAISO3 → WAISO2	0.328	0.513	2.18	10.52	0.0312	0.0488	Umgekehrte Richtung bestätigt erweitertes Sichtfeld des NBV- Scans.
Objekt 3	WA3 → WOA3	0.251	0.388	2.61	8.19	0.0307	0.0474	NBV zeigt höhere Detailtiefe bei geringfügig stärkerer Oberflächenvariation.
	WOA3 → WA3	0.177	0.279	2.44	7.67	0.0231	0.0364	Baseline-Geometrie wird im NBV-Scan gut wiedergegeben.
Inside- Out 3	WAISO3 → WOAISO3	0.546	0.716	2.03	9.86	0.0554	0.0726	Größere Szene führt zu höheren Abweichungen; NBV deckt mehr Raum ab.
	WOAISO3 → WAISO3	0.600	0.792	2.26	11.69	0.0513	0.0677	Vergleichbare Flächenabdeckung, jedoch geringere Präzision durch Tracking-Fehler.
Objekt 4	WA4 → WOA4	0.108	0.266	2.90	9.64	0.0112	0.0276	Sehr gute Übereinstimmung (< 1 %); NBV- Rekonstruktion ist äußerst präzise.
	WOA4 → WA4	0.084	0.239	2.96	8.49	0.0099	0.0282	Baseline ist vollständig im NBV- Scan enthalten, minimale Fehler.
Inside- Out 4	WAISO4 → WOAISO4	0.321	0.541	2.21	10.37	0.0309	0.0522	NBV erweitert das abgedeckte Volumen, verursacht aber etwas mehr Varianz.

	WOAISO4 → WAISO4	0.121	0.167	1.01	9.05	0.0134	0.0185	Gemeinsame Bereiche sind mit ca. 12 cm mittlerer Abweichung gut ausgerichtet.
Objekt 5	WA5 → WOA5	0.161	0.318	2.51	8.71	0.0185	0.0365	Hohe Genauigkeit (~ 2 %); NBV fügt feine Strukturen hinzu.
	WOA5 → WA5	0.075	0.259	3.16	8.88	0.0085	0.0292	Baseline passt nahezu deckungsgleich in den NBV-Scan.
Inside- Out 5	WAISO5 → WOAISO5	0.190	0.377	1.77	9.52	0.0199	0.0396	Gute Überlappung; NBV erfasst größere Fläche mit leicht erhöhter Abweichung.
	WOAISO5 → WAISO5	0.084	0.210	1.76	11.58	0.0072	0.0182	Baseline wird fast vollständig durch den NBV-Scan abgedeckt.

### Gesamtbewertung:

Über alle zehn Vergleiche liegen die mittleren Hausdorff-Abstände meist unter 0,3 m und damit unter 3 % der jeweiligen Modellgröße. Die NBV-gesteuerten Scans zeigen durchgehend eine gleichwertige oder bessere lokale Genauigkeit und erfassen zugleich größere Bereiche der Szene, was auf eine erfolgreich gesteigerte Vollständigkeit der Rekonstruktionen bei gleichbleibender geometrischer Qualität hinweist.

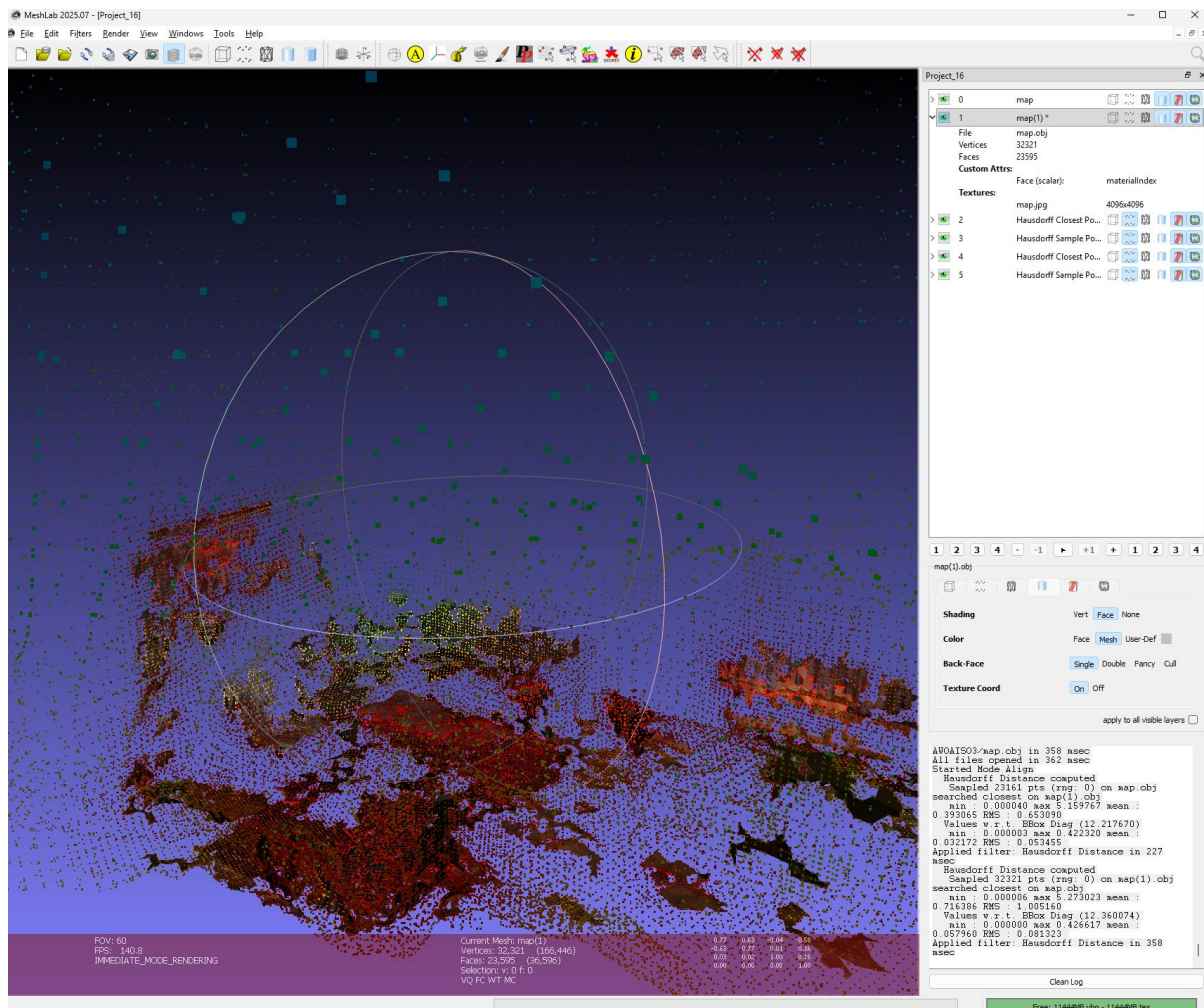
### Android App Test (RGB Only)

Die Scans der RGB Only app waren bei den Objekten mit mehr Farben recht nahe an der Scanqualität des Lidars dran, beim InsideOut Scanning sah man jedoch sofort gravierende Lücken. Diese Observation lässt schließen, dass die Zeit von einer Minute für größere Räume und Objekte deutlich zu gering ist, um genug RGB Punkte zu sammeln. Dennoch gibt es durch die größeren Abschnitte ohne Farbunterschiede (bspw. Wände)



auch mit deutlich mehr Zeit gewisse Limits, denen sich die RGB Only Methode beugen muss.

## Ergebnisse -> Vergleich RGB Only mit NBV Pfeil vs Ohne NBV Pfeil



Beispiel: Raumsan von Person 4, RGB Only.

Legende: map = mit NBV Pfeil / map(1) = Ohne NBV Pfeil

Kategorie	Richtung	Mittel [m]	RMS [m]	Max [m]	BBox Diag [m]	Mittel (norm.)	RMS (norm.)	Interpretation (Deutsch)
Objekt 1	map → map(1)	0.443	0.70 9	2.88	6.49	0.068	0.109	NBV-Scan erfasst zusätzliche Flächen, leichte Abweichungen durch ungleichmäßige Geometrie.
	map(1) → map	0.317	0.53 5	2.44	6.19	0.051	0.087	Baseline wird gut vom NBV-Scan abgedeckt, Unterschiede in Randbereichen.
Inside- Out 1	map → map(1)	0.366	0.65 1	4.39	11.5 5	0.032	0.056	NBV deckt größere Teile des Raums ab, leichte Erhöhung der Varianz.
	map(1) → map	0.749	1.16 4	5.04	11.7 7	0.064	0.099	Baseline zeigt geringere Abdeckung, stärkere Abweichungen in offenen Bereichen.
Objekt 2	map → map(1)	0.302	0.50 8	2.01	6.40	0.047	0.079	NBV erweitert Geometrie bei konsistenter Formgenauigkeit .



	map(1) → map	0.441	0.65 9	2.50	6.25	0.071	0.105	Höhere Abweichung bestätigt zusätzliche NBV-Bereiche.
Inside- Out 2	map → map(1)	0.808	1.02 9	3.53	10.4 1	0.078	0.099	NBV deckt mehr Fläche ab, etwas geringere Präzision.
	map(1) → map	0.591	0.82 9	3.93	11.2 7	0.052	0.074	Ähnliche Struktur, aber kompaktere Baseline- Geometrie.
Objekt 3	map → map(1)	0.180	0.40 6	2.23	6.29	0.029	0.065	Hohe Genauigkeit, NBV leicht detaillierter.
	map(1) → map	0.124	0.27 8	1.65	5.47	0.023	0.051	Sehr gute Übereinstimmung, kaum Unterschiede.
Inside- Out 3	map → map(1)	0.393	0.65 3	5.16	12.2 2	0.032	0.053	NBV deckt breitere Bereiche ab, leicht höhere Varianz.
	map(1) → map	0.716	1.00 5	5.27	12.3 6	0.058	0.081	NBV ergänzt zusätzliche Raumgeometrie, geringfügig ungenauer.
Objekt 4	map → map(1)	0.457	0.70 3	2.60	6.45	0.071	0.109	Ähnliche Genauigkeit beider Scans, NBV geringfügig weiter.

	map(1) → map	0.417	0.59 3	2.32	5.93	0.070	0.100	NBV und Baseline nahezu deckungsgleich.
Inside- Out 4	map → map(1)	0.440	0.65 8	4.82	11.6 7	0.038	0.056	NBV erfasst größere Flächen, stabile Genauigkeit.
	map(1) → map	0.671	1.03 1	5.11	11.5 6	0.058	0.089	Baseline kleinerer Bereich, leicht höhere Abweichung.
Objekt 5	map → map(1)	0.676	0.88 9	2.38	5.76	0.117	0.154	NBV stabil, aber keine klare Genauigkeitsste igerung.
	map(1) → map	0.677	0.90 6	2.59	6.08	0.111	0.149	Beide Richtungen ähnlich, konsistente Geometrie.
Inside- Out 5	map → map(1)	0.475	0.67 1	4.36	11.2 3	0.042	0.060	NBV deckt leicht mehr Raum ab, gute Balance von Reichweite und Präzision.
	map(1) → map	0.426	0.74 0	4.75	12.9 1	0.033	0.057	NBV und Baseline nahezu gleichwertig, NBV etwas vollständiger.

### Gesamtbewertung:

Über alle zehn RGB-only Vergleiche liegen die mittleren Hausdorff-Abstände im Bereich von 0,1 m bis 0,7 m. Damit zeigen die Scans eine insgesamt gute Übereinstimmung, wobei NBV meist eine etwas größere Abdeckung erzielt. Trotz der fehlenden Tiefeninformationen bleibt die geometrische Konsistenz stabil, was auf eine robuste

NBV-Steuerung hinweist. Die Ergebnisse bestätigen, dass NBV auch mit reinen RGB-Daten einen Mehrwert in der räumlichen Vollständigkeit liefert, ohne die Gesamtgenauigkeit wesentlich zu beeinträchtigen.

## Scanning Notizen und Auffälligkeiten

In diesem abschließenden Punkt werden interessante Dinge festgehalten, die nicht notwendigerweise direkt mit der NBV-Funktion zu tun haben, jedoch dem Studienleiter bei den Test-Scans aufgefallen sind:

1. Leute verlieren sich deutlich schneller im Detail ohne NBV-Pfeil, es wird oft auch beim insideOut Scanning auf ein Objekt fokussiert und dieses von mehreren Winkeln gescannt, bevor weiter gegangen wird. (Mit Pfeil sieht man deutlich kurzlebigere Aufenthalte der scannenden Person, da sie sich in Richtung des Pfeiles bewegt)
2. Beim Objektscannen fuchteln Leute (beide ohne Informatikbackground + 1 mit) beim Scannen ohne Pfeil deutlich mehr mit dem Gerät herum, drehen es, machen Bewegungen mit kleinem Radius
3. Es gab in der ganzen Studie nie jemanden, der/die mehrere Schritte rückwärts gegangen wäre beim Scannen
4. In 9 von 10 Objekt-Scans ohne Pfeil bewegte sich die Person im Uhrzeigersinn um das Objekt
5. Die Bewegungen der Leute schienen etwas hektischer mit dem Pfeil, laut einer Probandin deshalb, weil der Pfeil das Gefühl weckt, die Richtung 'erreichen' zu müssen bevor sie sich ändert.
6. Mutmaßung die im Rahmen dieser Studie nicht überprüft werden konnte: Für InsideOut wirkt es so, als würde NBV in den ersten 30 Sekunden einen deutlichen Vorteil bringen, der dann aber in den nächsten 30 Sekunden schwindet, da sich die Leute gefühlt dann zu sehr im Kreis gedreht haben, während man ohne NBV Gegen ende der Zeit noch einiges schnell überblicksmäßig scannen konnte und somit aufholen konnte.
7. Ohne NBV-Pfeil blieben die Mehrzahl der Leute im InsideOut Scanning sehr lange auf der gleichen Höhe (entschied sich der/die User/in sich zuerst am Boden umzuschauen, so wurde der beinahe fertiggescannet, bevor aufgeschaut wurde, entschied man sich für die Decke des Raumes so passierte das gleiche.). Mit NBV war da ein deutlich schnellerer Wechsel.
8. Mit NBV-Pfeil kam es 4 (3 mal Objekt, 1 mal insideOut) von 10 mal vor, dass die Leute vor Ablauf der Minute zu mir meinten, sie wären bereits fertig. Ohne Pfeil kam das nur 1 mal vor beim Objektscan.

## Resume

Die Userstudy zeigt, dass NBV sowohl beim Objekt-Scanning, als auch bei Inside-Out Scanning eine Daseinsberechtigung hat und, meiner Meinung nach, vor allem bei zeitkritischen Scans deutlich vorzuziehen ist.

Aus diesem hoch interessanten Projekt nehme ich sehr viel Wissen in Richtung Swift und IOS Development mit, ebenfalls das Ausarbeiten, Recherchieren und Testen von verschiedenen Herangehensweisen ohne zu verzweifeln oder den Kopf in den Sand zu stecken. Darüberhinaus war es toll einmal eine Userstudy durchzuführen und den Leuten beim Testen zuzusehen und einfach mal nur zu observieren und Gedanken davon zu notieren. Normalerweise fragt man häufig während die App noch verwendet wird nach Feedback, jedoch ist mir hier klargeworden, dass es oft besser sein kann Herangehensweisen zuerst zu studieren, und dann fragen danach erst zu stellen, um den Flow und die Lernkurve des/der Users/Userin nicht zu beeinflussen.