1 Stand der Technik

SIFT [**?**] ist der wohl bekannteste Ansatz und wird häufig als Goldstandard verwendet, um Verfahren miteinander zu vergleichen. SIFT hat allerdings den Nachteil, dass es zu langsam für Echtzeitanwendungen ist. SIFT ermittelt die Kandidaten-Punkte im Kern mittels des Difference of Gaussians Verfahrens (DoG), welches aus drei Schritten besteht. Im ersten Schritt wird auf das Bild in verschieden starken Stufen ein Weichzeichner (diskreter Gauß-Filter) angewandt. Damit liegt das Bild mit zunehmend niedrigerem Kontrast vor. Zwischen den einzelnen Kontraststufen wird die Differenz ermittelt. Liegen die Differenzen vor, kann nach Extrempunkten in den Differenzen gesucht werden. Hierbei wird jedes Pixel im Bild mit seinen direkten Nachbarn und denen in der vorherigen und nachfolgenden Stufe verglichen. Ist das Pixel ein Extrempunkt, wird es als Kandidat gewählt. DoG wird auf mehreren Skalierungen des Bildes angewandt, um Skalierungs-Invarianz zu erreichen. Im Anschluss werden die Kandidaten gefiltert. Kandidaten, die einen niedrigen Intensitätswert haben oder eine Kante darstellen, werden hierbei ausgeschlossen. Sind die Positionen der Features bekannt, kann der Feature-Vektor f für den Punkt p ermittelt werden. Um den Deskriptor für den Punkt p zu berechnen werden zwei Größen benötigt. Zum einen die Skalierung, in der der Punkt ermittelt wurde - diese ist bereits bekannt - und zum anderen die Orientierung des Punktes. Um diese zu bestimmen werden in einem Bereich um den Punkt p die Gradienten berechnet und in ein Histogramm abgetragen. Der Histogramm-Eintrag (Bin) mit dem größten Wert entscheidet die Orientierung des Punktes. Sind Skalierung und Orientierung bekannt, wird auf Basis der beiden ein Bereich gewählt und in einen 16×16 großen Bereich skaliert. Dieser Bereich wird in 16 gleichgroße Bereiche unterteilt, für die jeweils ein Histogramm gebildet wird. Die Bins der Histogramme beschreiben in 8 diskreten Stufen die Orientierung der Punkte im jeweiligen Bereich. Die einzelnen Punkte tragen die Stärke des Gradienten im jeweiligen Bin des Histogramms ab. Diese Histogramme bilden den Feature-Vektor f, der für den Vergleich genutzt wird.

SURF [**?**] ist ähnlich aufgebaut wie SIFT, nur wird der genutzte Gauß-Filter durch Box-Filter, die besonders effizient berechnet werden, approximiert. Des Weiteren wird der Feature-Vektor anders gebildet. Mit beiden Anpassungen wird eine bessere Leistung erreicht. Bei der Approximation des Gauß-Filters wird zuerst das Integral-Bild I_Σ (Gleichung 1) aus den Intensitätswerten I des Bildes berechnet.

$$I_\Sigma(x, y) = \sum_{i=0}^{i \leq x} \sum_{j=0}^{j \leq y} I(i, j) \tag{1}$$

Ist I_Σ bekannt, kann die Intensitäts-Summe eines beliebigen rechteckigen Ausschnitts sehr effizient berechnet werden. Hierzu werden nur die Werte aus I_Σ für die Eckpunkte P_1 - P_4 eines Ausschnittes D benötigt. Die Fläche für D lässt sich dann mit $f(D)$ (Gleichung 2) effizient berechnen.

$$f(D) = I_\Sigma(x_{P_1}, y_{P_1}) + I_\Sigma(x_{P_4}, y_{P_4}) - I_\Sigma(x_{P_2}, y_{P_2}) - I_\Sigma(x_{P_3}, y_{P_3}) \tag{2}$$

Um den Gauß-Filter mit Box-Filtern zu approximieren, wird die Determinante der Hesse-Matrix genutzt. Um die approximierte Hesse-Matrix für einen Punkt p mit der

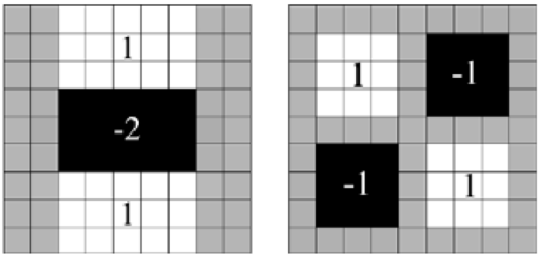

Abbildung 1: Box-Filter für zweite Ableitung nach y (links) und Ableitung nach xy (rechts). [**?**]

Skalierung s zu bilden, muss man die zweite Ableitung der Gauß-Funktion nach x, y und xy mit Box-Filtern approximieren (Siehe Gleichung 3).

$$H_{approx.}(p,s) = \begin{pmatrix} D_{xx}(p,s) & D_{xy}(p,s) \\ D_{xy}(p,s) & D_{yy}(p,s) \end{pmatrix} \tag{3}$$

Die Determinante der Hesse-Matrix kann dann wie in Gleichung 4 berechnet werden, wobei $w \approx 0{,}9$ ist und den Fehler der Approximation minimieren soll.

$$det(H_{approx.}) = D_{xx}D_{yy} - (wD_{xy})^2 \tag{4}$$

Ein Beispiel für die Box-Filter ist in Abbildung 1 zu sehen. Sowohl die verschiedenen Gauß-Stufen als auch die verschiedenen Skalierungs-Stufen werden durch ein Vergrößern der Filter-Maske erreicht. Der Deskriptor wird in zwei Schritten berechnet. Zuerst wird die Hauptorientierung des Punktes berechnet. Dann wird auf dieser und der Skalierung basierend ein Bereich in dem Bild gewählt und in 16 Bereiche unterteilt. Für jeden der 16 Bereiche wird die Intensität der Punkte in einem 4 Bin großen Histogramm abgetragen, was zu einem Feature-Vektor der Größe 64 führt.

ORB [**?**] geht diesen Nachteil an, indem zuerst das Massezentrum des Bildes C (Gleichung 6) und dessen Orientierung θ (Gleichung 7) über die Momente m_{00}, m_{10} und m_{01} (Gleichung 5) berechnet wird. Mit diesen Parametern werden die Koordinaten der Paare umgeformt, um so der Anfälligkeit für Rotation entgegen zu wirken. Des Weiteren liefert ORB eine gelernte Paar-Auswahl von 256 Paaren, die dahingehend optimiert ist, dass die Paare unkorreliert sind und eine hohe Varianz aufweisen.

$$m_{pq} = \sum_{x,y} x^p y^q I(x,y) \tag{5}$$

$$C = (\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}}) \tag{6}$$

$$\theta = atan2(m_{01}, m_{10}) \tag{7}$$

Um aus den korrespondierenden Bildpaaren auf die Positionen der Kameras zu schließen, wird die Fundamental-Matrix $F = K_1^{-1T} E K_2^{-1}$ für jedes Bildpaar ermittelt. K_1

und K_2 sind die beiden Kalibrierungs-Matrizen der Kameras und E ist die Essential-Matrix, welche die Bewegung und Rotation der Kamera beschreibt. Ist die Kalibrierung der Kamera bekannt, kann direkt E berechnet werden. F beschreibt die Kamera-Bewegung und Rotation von einem zum anderen Bild, hat den Rang 2 und bezieht dabei die Kalibrierung der Kameras mit ein. Um F zu ermitteln werden mindestens acht korrespondierende Features $(p_{A1}, p_{B1})...(p_{A8}, p_{B8})$ zwischen den Bildern A und B eines Paares benötigt. Die Bildpunkte p_{Ai} und p_{Bi} der Paare sind dabei nicht in Pixel-Koordinaten, sondern relativ zum Bild-Ebenen-Zentrum angegeben. Die Koordinaten liegen also im Bereich $[-1, 1]$. Da es sich um homogene Koordinaten handelt, wird neben dem x und y Wert noch eine Eins hinzugefügt. Hierzu werden mindestens acht Gleichungen der Form $p_{Ai}^T F p_{Bi} = 0$ (Siehe auch Gleichung 8) gebildet und daraus ein Gleichungssystem der Form $Af = 0$ erzeugt, wobei f die Unbekannten aus F enthält.

$$\begin{aligned} x_{p_{Ai}} x_{p_{Bi}} f_1 + x_{p_{Ai}} y_{p_{Bi}} f_2 + x_{p_{Ai}} f_3 + \\ y_{p_{Ai}} x_{p_{Bi}} f_4 + y_{p_{Ai}} y_{p_{Bi}} f_5 + y_{p_{Ai}} f_6 + \\ x_{p_{Bi}} f_7 + y_{p_{Bi}} f_8 + f9 = 0 \end{aligned} \tag{8}$$

Das Gleichungssystem wird mittels Singulärwertzerlegung (SVD) gelöst. Man erhält drei Matrizen U, S und V. U hat die Größe $m \times m$, S die Größe $m \times 9$ und V die Größe 9×9. S ist eine Diagonal-Matrix, deren Einträge die Singulärwerte von A in absteigender Reihenfolge enthalten. Aus der letzten Spalte von V (Siehe Gleichung 9), welche den kleinsten Singulär-Vektor enthält, kann die Approximation $\hat{F}$ (Siehe Gleichung 10) von F rekonstruiert werden.

$$V[9] = \begin{bmatrix} \hat{f}_1 & \hat{f}_2 & \hat{f}_3 & \hat{f}_4 & \hat{f}_5 & \hat{f}_6 & \hat{f}_7 & \hat{f}_8 & \hat{f}_9 \end{bmatrix} \tag{9}$$

$$\hat{F} = \begin{bmatrix} \hat{f}_1 & \hat{f}_2 & \hat{f}_3 \\ \hat{f}_4 & \hat{f}_5 & \hat{f}_6 \\ \hat{f}_7 & \hat{f}_8 & \hat{f}_9 \end{bmatrix} \tag{10}$$

Hierbei wird der Vektor der Länge 9 in drei Teile zerlegt, welche die Zeilen von $\hat{F}$ bilden. Da durch Rauschen $\hat{F}$ den Rang 3 haben kann, wird von $\hat{F}$ noch einmal die SVD gebildet und in S das letzte Diagonal-Element explizit auf 0 gesetzt, was hier als Matrix $\hat{S}$ beschrieben wird. Damit kann die Approximation von F neu berechnet werden ($\hat{F} = U\hat{S}V^T$) und hat damit Rang 2. Der Unterschied bei der Berechnung der Essential-Matrix zur Fundamental-Matrix ist, dass zum einen bei der Korrektur der Approximation neben dem letzten Eintrag der Diagonal-Elemente, der auf 0 gesetzt wird, die beiden verbleibenden Diagonal-Elemente auf 1 gesetzt werden. Die daraus resultierende Matrix wird im Folgenden mit D bezeichnet. Des Weiteren wird bei den homogenen Bildpunkten nicht 1 gesetzt, sondern c, die Kamerakonstante. Ist E bekannt, kann die Kamera-Pose, bestehend aus dem Kamera Zentrum c und der Kamera-Rotation R, geschätzt werden [**?**]. Hier ergeben sich vier mögliche Lösungen, die in Gleichung 11 zu sehen sind. Mit W, wie in Gleichung 12 beschrieben.

$$
\begin{aligned}
c_1 &= U(:,3), & R_1 &= UWV^T \\
c_2 &= -U(:,3), & R_2 &= UWV^T \\
c_3 &= U(:,3), & R_3 &= UW^TV^T \\
c_4 &= -U(:,3), & R_4 &= UW^TV^T
\end{aligned}
\tag{11}
$$

$$
W = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{12}
$$

Um zu ermitteln, welche der vier Konfigurationen die richtige ist, kann geprüft werden, ob die rekonstruierten 3D-Punkte vor beiden Kameras liegen. Es wird die Konfiguration gewählt, bei der am meisten Punkte vor den Kameras liegen. Sind nun die ersten beiden Kamera-Posen bekannt, können weitere Kameras mittels Perspektive-n-Points-Algorithmus (PnP) [**?**] hinzugefügt werden. Hierbei wird anhand der bereits ermittelten 3D-Punkte die Pose einer weiteren Kamera mittels Least-Square-Fit gefunden. Da die Pose der Kamera 6 Freiheitsgrade hat, werden mindesten 6 korrespondierende Punkte benötigt. Wurden alle Kamera-Posen und 3D-Punkte geschätzt, müssen diese noch einmal korrigiert werden. Diesen Schritt nennt man Bundle-Adjustment (BA) [**?**]. Beim BA wird für jede der m Kamera-Posen A mit den Parametern a_j jeder der n 3D-Punkte P_i, der von der Kamera aufgenommen wurde, zurück auf das Kamera-Bild projiziert und der Abstand des so ermittelten Bild-Punktes zum originalen Bild-Punkt aus der Feature-Detektion gemessen und minimiert. Das Optimierungsproblem kann, wie in Gleichung 13 zu sehen, formuliert werden.

$$
\min_{a_j, P_i} = \sum_{i=1}^{n} \sum_{j=1}^{m} v_{ij} d(Q(a_j, P_i), x_{ij}) \tag{13}
$$

Hierbei gilt $v_{ij} = 1$, wenn der Punkt i von Kamera j erfasst wurde, ansonsten 0. $Q(a_j, P_i)$ ist die geschätzte Projektion des Punktes P_i auf das Bild der Kamera j, und d ist die Euklidische Distanz. x_{ij} ist der detektierte Punkt auf dem Bild.

1.1 Registrierung von 3D-Punktwolken

Bei der Registrierung von 3D-Punktwolken probiert man eine Transformationen T zu finden, die eine Quell-Punktwolke P_s so transformiert, dass der Abstand $d(P_s, P_t)$ zu einer Ziel-Punktwolke P_t minimiert wird. Das Abstandsmaß $d(P_s, P_t)$ kann verschieden formuliert werden. Eine Möglichkeit besteht darin, die Summe über die Abstände eines jeden Punktes in P_s zum nächsten Punkt in P_t zu bilden und dies als Maß $d(P_s, P_t)$ zu nutzen. Dies kann als Optimierungsproblem (Gleichung 14) formuliert werden.

$$
\underset{R,t}{\operatorname{argmin}} \left(\sum_{i=1}^{N} \| R p_{s_i} + t - p_{t_i} \|^2 \right) \tag{14}
$$

Hierbei sind R und t die gesuchte Rotation und Translation und p_{t_i} und p_{s_i} bilden ein korrespondierendes Punktpaar aus den beiden zu registrierenden Punktwolken. N ist die Anzahl Punkte in der Quell-Punktwolke.

Die am weitesten verbreitete Methode zur Registrierung von 3D Punktwolken ist Iterative Closest Points (ICP) [**?**]. ICP basiert auf dem Ansatz, dass zwei Punktwolken iterativ aneinander angenähert werden, um den Abstand zwischen den Punktwolken schrittweise zu minimieren. ICP approximiert die Lösung des Problems in Gleichung 14, indem iterativ zwei Schritte durchgeführt werden: Erst werden die Zuordnungen für jeden Punkt in der Quell-Punktwolke zu einem Punkt in der Ziel-Punktwolke gesucht und danach wird die Rotation R und Translation t geschätzt, die die zugeordneten Punkte bestmöglich annähert. Um eine Zuordnung zwischen den Punkten zu erreichen, wird für jeden Punkt p_i aus der Quell-Punktwolke P_s der Punkt aus der Ziel-Punktwolke gesucht, der p_i am nächsten ist. Im zweiten Schritt wird zuerst für beide Punktwolken das Massezentrum c_s und c_t berechnet. Hierbei wird das Massezentrum c einer Punktwolke P mit n Punkten, wie in Gleichung 15 gezeigt, berechnet.

$$c = \frac{1}{n} \sum_{i=1}^{n} p(i) \tag{15}$$

Mit c_s und c_t lässt sich die Kreuzkovarianz-Matrix H (Gleichung 16) der beiden Punktwolken berechnen, auf der die SVD ($svd(H) = UDV^T$) angewandt wird.

$$H = \sum_{i=1}^{n} (p_{t_i} - c_t)(p_{s_i} - c_s)^T \tag{16}$$

Sind U und V bekannt, kann damit die Rotation $R = VU^T$ und mit R die Translation $t = c_t - Rc_s$ berechnet werden. Da die Zuordnung im ersten Schritt nicht ideal sein muss, wird der Vorgang solange wiederholt, bis es in der Transformation T, bestehend aus R und t, zu keinen größeren Änderungen mehr kommt. Das ist der Fall, wenn T etwa der Einheitsmatrix entspricht.

Ein weiteres Problem bei den meisten ICP-Ansätze ist das Fehlen einer Schätzung für die Skalierung der zu registrierenden Punktwolke. Diese ist aber nötig, da bei SFM keine Information über die reale Ausdehnung eines Objektes ermittelt werden kann. Es gibt einige wenige Ansätze, die auch eine Schätzung für die Skalierung ermitteln [**?**]. In [**?**] wird die Skalierung geschätzt, nachdem die Rotation R bekannt ist, also nach der SVD. Ist R bekannt, können die Vektoren s und t gebildet werden (Siehe Gleichung 17).

$$s_i = R(p_{s_i} - \bar{p_s}), \quad t_j = p_{t_j} - \bar{p_t} \tag{17}$$

Mittels dieser Vektoren kann die Skalierung $\hat{s}$ berechnet werden. Die Berechnung ist in Gleichung 18 zu sehen.

$$\hat{s} = \sum_{i,j} t_j^T s_i / \sum_i s_i^T s_i \tag{18}$$

Die Translation muss auch in angepasster Form berechnet werden, wie in Gleichung 19 zu sehen ist.

$$t = \bar{p_t} - \hat{s} R \bar{p_s} \tag{19}$$

Damit wird das Optimierungs-Problem in Gleichung 14 umformuliert, wie in Gleichung 20 zu sehen ist.

$$\underset{M,R,t}{\operatorname{argmin}}(\sum_{i=1}^{N}\sum_{k=1}^{K} m_{jk}(\|Rp_{s_i} + t - p_{t_k}\|_2^2 - \alpha)) \tag{20}$$

Dabei ist M die Gewichtungsmatrix für die Zuordnungen zwischen den Quell- und Zielpunkten. Der Parameter α ist dazu da, um Korrespondenzen zwischen schlechten Paaren zu bestrafen. Die Idee dabei ist, dass Distanzen unter dem Schwellwert α die Kosten senken und Distanzen über α die Kosten erhöhen. Die Matrix M wird in jeder Iteration wie in Gleichung 21 initialisiert.

$$m_{ij} = e^{-\beta(\|Rp_{s_i}+t-p_{t_j}\|_2^2-\alpha)} \tag{21}$$

Dabei ist β ein Parameter, der über die Iteration erhöht wird.

Im Vergleich zu RPM werden die räumlichen Entfernungen bei der Berechnung von M durch hybride Feature-Distanzen ersetzt (Siehe Gleichung 22).

$$m_{ij} = e^{-\beta(\|\hat{f}_{s_i}-f_{t_j}\|_2^2-\alpha)} \tag{22}$$

$\hat{f}_{s_i}$ ist der hybride Feature-Vektor für den aus der vorherigen Iteration transformierten Punkt p_{s_i} und f_{t_j} ist der Feature-Vektor für den Punkt p_{t_j}. Des Weiteren werden die Parameter α und β in jeder Iteration durch ein neuronales Netz geschätzt. β wird dabei nicht zwangsläufig wie in RPM immer weiter erhöht. Zuletzt wird, wie bei ICP, die SVD der Punktpaare ermittelt und so die Rotation gefunden, nur fließt beim Bilden der SVD auch das Gewicht der einzelnen Paare mit ein. Die Transformation kann so wie bei ICP ermittelt werden.

2 Realisierung

2.1 Umsetzung Segmentierung

Die Hauptkrümmung eines Punktes p_i kann über die Normale des Punktes n_i und die Normalen der k nächsten Nachbarpunkte $n_j \in P_i$ berechnet werden.

$$m_j = (I - n_i \otimes n_i) \cdot n_j \tag{23}$$

Aus den Abbildungen m_j (siehe Gleichung 23) für alle P_i kann die Kovarianzmatrix C_i berechnet werden, wie in Gleichung 24 zu sehen ist.

$$C_i = \frac{1}{k} \sum_{j=1}^{k} (m_j - \bar{m}) \otimes (m_j - \bar{m}) \tag{24}$$

Die Hauptkrümmung kann aus den Eigenwerten $0 \leq \lambda_1 \leq \lambda_2 \leq \lambda_3$ von C_i bestimmt werden. λ_3 ist die stärkste Krümmung und λ_2 die schwächste Krümmung.

In den Gleichungen in 25 sind einige der untersuchten Ansätze für eine Funktion $k(p_i)$ aufgeführt.

$$\begin{aligned} k_1(p_i) &= \lambda_3 \\ k_2(p_i) &= \lambda_2 \\ k_3(p_i) &= (\lambda_3 + \lambda_2)/2 \end{aligned} \tag{25}$$

Je höher der Wert der Funktion $k(p_i)$ aus den Gleichungen in 25 ist, desto wahrscheinlicher gehört ein Punkt zum Stiel einer Pflanze. Die Entscheidungsregel dafür zeigt Gleichung 26. Ein guter Wert für den Schwellwert T_k muss in Experimenten für die Funktionen k_1, k_2 und k_3 gefunden werden. Diese Experimente haben gezeigt, dass k_1 die besten Ergebnisse liefert.

$$f(p_i) = \begin{cases} 1 & k(p_i) \geq T_k \\ 0 & \text{sonst} \end{cases} \tag{26}$$

2.2 Verbesserung des Ergebnisses

$$\begin{aligned} &L = \{0{,}1{,}2\} \\ &x \in L \\ &w(x) = \begin{cases} 0{,}5 & x = 2 \\ 1 & \text{sonst} \end{cases} \\ &H_i(x) = \textstyle\sum_{j=0}^{k} (w(x) | N_{ij} = x) \\ &s_i = argmax_x(H_i(x)) \end{aligned} \tag{27}$$

3 Ergebnisse

3.1 Vergleich von Verfahren zur Segmentierung von Pflanzen auf 3D-Punktwolken

Datenbasis für das Training von PointNet++ IoU wird über alle zu segmentierenden Klassen gemittelt. Für jede Klasse C wird dazu der Jaccard-Koeffizient $j_c(\hat{P}_c, P_c) = \frac{|\hat{P}_c \cap P_c|}{|\hat{P}_c \cup P_c|}$ gebildet. $\hat{P}_c$ ist die Menge an Punkten aus der Schätzung des Netzes, welche als C klassifiziert werden. P_c ist die Menge der als C klassifizierten Punkte aus dem Groundtruth. Die angegebenen IoU-Werte sind über alle Klassen gemittelt (mIoU).