

Studienarbeit

Anwendung des SURF-Algorithmus

Flugzeugerkennung in Satellitenbildern



Felice Serra
Mtrklnr.: 2944562
felice.serra@gmx.de
Wilhelm-Schickard-Institut für Informatik
Lehrstuhl für Graphisch-Interaktive Systeme
Prof. Dr. Schilling

WS 2014/2015



Inhaltsverzeichnis

1. Übersicht	1
2. Theorie	2
2.1. Der SURF-Algorithmus	2
2.1.1. Finden der <i>Points Of Interest</i>	3
2.1.2. Erzeugung der Deskriptoren	3
2.2. Bildung der Merkmals-Paare	4
2.2.1. Brute-force	4
2.2.2. FLANN	4
2.3. Finden der perspektivischen Transformation	5
2.3.1. Reguläre Methode	5
2.3.2. RANSAC-basierte robuste Methode	5
2.3.3. Least-Median robuste Methode	6
2.4. Der Canny-Algorithmus	6
2.4.1. Glättung:	6
2.4.2. Berechnen der partiellen Ableitungen:	6
2.4.3. <i>Non-maximum Supression</i> :	6
2.4.4. Anwendung der Schwellwerte:	7
3. Vorbereitung	8
3.1. Das Objekt	8
3.2. Die Szene	9
3.3. Graphische Benutzeroberfläche	10
4. Anwendung	12
4.1. Verwendete Software	12
4.2. Ablauf	12
4.2.1. Erstellen der Graustufenbilder	12
4.2.2. Anwendung des Canny-Algorithmus	12
4.2.3. Anwendung des SURF-Algorithmus	13
4.2.4. Bildung der Merkmals-Paare	13
4.2.5. Schwellwert-Filter	13
4.2.6. Perspektivische Transformation	14
4.2.7. Anwendung der perspektivischen Transformation	14



5. Auswertung	15
5.1. Parameterwahl	15
5.1.1. Zoom-Stufe	15
5.1.2. Hintergrundfarbe des Objekts	16
5.1.3. Canny Schwellwerte	17
5.1.4. Maximale Distanz der Merkmals-Paare	17
5.1.5. RANSAC Schwellwert	18
5.1.6. Graustufen oder Canny	19
5.1.7. Zusammenfassung	19
6. Fazit und kritische Bewertung	20
Literaturverzeichnis	21
A. Anhang	i
A.1. Eidesstattliche Erklärung	i
A.2. Adresse	i



1. Übersicht

Ziel dieser Studienarbeit ist die Anwendung des SURF(*Speeded-Up-Robust-Feature*)-Algorithmus zur Objekterkennung. Dabei werden markante Merkmale aus Bildern extrahiert und beschrieben. Die Aufgabe ist es, Flugzeuge in Satellitenbildern zu erkennen. Der SURF-Algorithmus liefert Merkmale des Flugzeuges, die in der Szene gesucht und verglichen werden. Das Projekt ist in Java geschrieben und nutzt die OpenCV-Bibliotheken zur Bildbearbeitung und Objekterkennung. Die Satellitenbilder stammen von Google-Maps.

Zuerst wird ein Überblick über die verwendeten Algorithmen zur Bildverarbeitung und Objekterkennung gegeben. Anschließend folgt eine Beschreibung der Vorbereitung, die Verwendung der Google-Maps-API und die Erstellung einer graphischen Benutzeroberfläche. Danach wird auf den Ablauf der eigentlichen Objekterkennung eingegangen. Abschließend werden die erhaltenen Ergebnisse präsentiert und bewertet.



2. Theorie

Dieses Kapitel gibt einen Überblick über die in diesem Projekt genutzten Algorithmen. Zur Objekterkennung dient der SURF-Algorithmus. Er erkennt und beschreibt markante Merkmale eines Bildes und ist eine Weiterentwicklung des SIFT-Algorithmus. Des Weiteren werden die verschiedenen Möglichkeiten zur Bildung des besten Merkmal-Paares behandelt. Außerdem werden Algorithmen für die Erzeugung der perspektivischen Transformation zwischen Objekt-Merkmalen und Szene-Merkmalen vorgestellt. Zur Bildverarbeitung wird der Canny-Algorithmus erläutert. Er liefert eine Methode zur Kantenerkennung.

2.1. Der SURF-Algorithmus

SURF (Speeded Up Robust Features) ist ein robuster Algorithmus zur Merkmals-Erkennung und Beschreibung. Er findet Anwendung z.B. in der Objekterkennung oder der 3D-Rekonstruktion. Das erste Mal vorgestellt wurde er von Herbert Bay et al. 2006 [Bay u. a. 2006]. Er übernimmt Ansätze des SIFT-Algorithmus, ist aber deutlich schneller als dieser und gilt auch als robuster. Der Algorithmus läuft in zwei Schritten ab. Zuerst werden markante Merkmale eines Bildes lokalisiert, sogenannte *Points Of Interest*. Diese werden im zweiten Schritt mit Deskriptoren versehen, die diese Merkmale näher beschreiben.

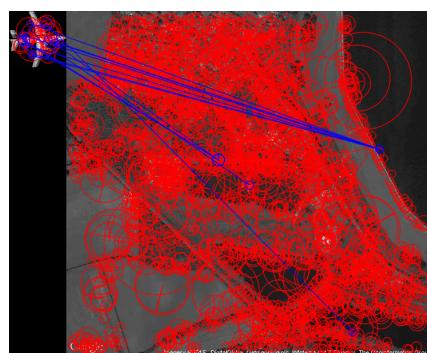


Abbildung 2.1.: Beispiel für SURF-Merkmale in unterschiedlicher Größe und Orientierung.



2.1.1. Finden der *Points Of Interest*

Points of Interest sind markante Punkte eines Bildes, wie z.B. Ecken, Blobs (zusammenhängende Bereiche gleicher Intensität) oder T-Kreuzungen von Kanten. Bei der Suche nach den *Points of Interest* wird ein Blob-Detektor basierend auf der Hesse-Matrix angewandt. Dabei werden Blob-ähnliche Strukturen dort gesucht, wo die Determinante der Hesse-Matrix am größten ist. Einer der Hauptgründe für die schnellere Ausführung ist die Verwendung sogenannter Integralbilder [Viola und Jones 2001]. Ein Wert an den Koordinaten (x, y) im Integralbild entspricht der Summe der Pixel in einem Rechteck vom Ursprung bis zum Punkt (x, y) im Originalbild. Durch die Verwendung von Integralbildern lässt sich die Summe der Pixel in einem Rechteck in nur drei Additionen berechnen. Damit ist die schnelle Anwen-

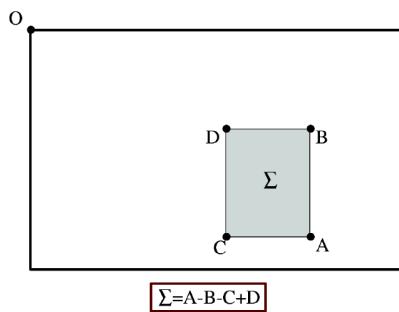


Abbildung 2.2.: Es braucht nur drei Additionen und vier Speicherzugriffe, um die Summe der Werte innerhalb des Rechtecks zu bilden. [Bay 2008]

dung von unterschiedlich großen, rechteckigen Filtern möglich, um die sogenannte *scale-space*-Repräsentation zu erhalten. Die unterschiedlichen Stufen werden in einem Pyramiden-Bild gespeichert. Diese Art des Findens von *Points Of Interests* hat den Vorteil der Wiederholbarkeit. Das heißt der Detektor findet den selben Punkt auch in unterschiedlichen Perspektiven, z.B. nach Rotation oder Skalierung des Bildes.

2.1.2. Erzeugung der Deskriptoren

Ein Deskriptor ist eine Beschreibung, die einem *Point Of Interest* hinzugefügt wird. Diese Beschreibung enthält Informationen über die Intensitätsverteilung der Pixel in der Nachbarschaft. SURF berechnet diese in zwei Schritten. Zuerst wird eine kreisförmige Region um den *Point Of Interest* nach markanten Blobs abgesucht. Mit diesen Informationen wird die Orientierung des *Point Of Interest*s festgelegt. Im zweiten Schritt wird ein quadratischer Bereich gebildet, der der vorher festgelegten Orientierung folgt. Dieser Bereich wird wieder in kleinere Rechtecke unterteilt und der SURF-Deskriptor wird erzeugt, indem man *Haar-Wavelet-Responses* [Haar 1910]



in x- und y-Richtung berechnet. Die Aufsummierung der einzelnen *Haar-Wavelet-Responses* in einem Vektor ergibt den SURF-Deskriptor des *Point Of Interests*.

2.2. Bildung der Merkmals-Paare

Hat man eine Menge an Merkmalen eines Objektes und einer Szene extrahiert, sucht man nach passenden Merkmals-Paaren. Die Fragestellung ist, welches Objekt-Merkmal ist welchem Szene-Merkmal am ähnlichsten. Ähnlichkeit meint hierbei die Differenz der Merkmals-Deskriptoren, die vom SURF-Algorithmus erzeugt wurden. Da die Menge der Objekt-Merkmale groß sein kann und, im Falle der Objekterkennung, die Menge der Szene-Merkmale klassischer Weise noch größer ist, braucht es Algorithmen zur effektiven Berechnung der Merkmals-Paare. Dies spielt vor allem bei zeitkritischen Problemen eine Rolle. In OpenCv stehen folgende *DescriptorMatcher* zur Verfügung:

2.2.1. Bruteforce

Die Bruteforce-Methode berechnet für ein gegebenes Objekt-Merkmal die Distanzen zu allen Szene-Merkmalen. Das Szene-Merkmal mit der geringsten Distanz wird zurückgegeben. So verfährt man mit allen Objekt-Merkmalen. Entstehen dabei Merkmals-Paare mit dem selben Szene-Merkmal, wird das Merkmals-Paar mit der geringeren Distanz beibehalten. Das andere Merkmals-Paar muss neu berechnet werden. Es werden vier Typen von Bruteforce-Methoden angeboten: Bruteforce-L1, Bruteforce-L2, Hamming und Hamming(2). Sie unterscheiden sich in der Metrik zur Berechnung der Distanz der Merkmals-Deskriptoren. Für SURF-Deskriptoren wird Bruteforce-L2 empfohlen. Diese Methode wird auch in diesem Projekt verwendet.

2.2.2. FLANN

FLANN steht für *Fast Library for Approximate Nearest Neighbors* [Muja und Lowe 2014]. Dies ist eine Bibliothek für schnelle, approximierte *Nearest-Neighbor*-Suchen in mehrdimensionalen Räumen. Für ein gegebenes Problem wird automatisch der beste Algorithmus mit den besten Parametern ausgewählt. Dies ist eine schnelle Methode für zeitkritische Anwendungen, denn es wird, im Gegensatz zur Bruteforce-Methode, hier nur eine Approximation der optimalen Lösung berechnet. Da in diesem Projekt keine Notwendigkeit für eine Beschleunigung der Merkmals-Bildung besteht, wird auf die Anwendung der FLANN-Methode verzichtet.



2.3. Finden der perspektivischen Transformation

Ein weiteres Problem der Objekterkennung ist die Berechnung der perspektivischen Transformation, die die Punktmenge der Objekt-Merkmale auf die Punktmenge der Merkmale eines, in der Szene gefundenen, Objektes abbildet. Damit kann die Position und Orientierung eines Treffers in der Szene bestimmt werden. In OpenCv steht dafür die Funktion *findHomography* zur Verfügung. Folgende Algorithmen zur Bestimmung der perspektivischen Transformation können gewählt werden:

2.3.1. Reguläre Methode

Bei der regulären Methode werden alle Punkte berücksichtigt. Die Funktion findet die perspektivische Transformation zwischen der Objekt-Menge und der Szenen-Menge, die den Rückprojektionsfehler minimiert. Der Rückprojektionsfehler beschreibt die Diskrepanz zwischen der Ausgangsmenge und der transformierten Menge. Die Funktion berechnet eine Schätzung der Homographie mittels einer einfachen Methode der kleinsten Quadrate (*Least-Square*). Diese wird in OpenCv anschließend mit der Levenberg-Marquardt-Methode optimiert. Bei Problemfällen mit vielen Ausreißern liefert die reguläre Methode eher schlechte Ergebnisse. In diesem Fall sollte eine der robusten Methoden gewählt werden.

2.3.2. RANSAC-basierte robuste Methode

RANSAC (*Random Sample Consensus*) [Fischler und Bolles 1981] beschreibt eine Methode, bei der Ausreißer bestimmt werden, die nicht in die Berechnung mit einfließen. Dabei werden nur so viele Punkte zufällig ausgewählt, wie für die Berechnung einer Transformation nötig sind. Von diesem ersten Modell wird der Rückprojektionsfehler berechnet. Ist dieser unter einem, vom Anwender definierten, Schwellwert, werden die genutzten Punkte in das sogenannte Consensus-Set aufgenommen. Ist der Fehler oberhalb des Schwellwertes werden die Punkte als Ausreißer klassifiziert. Diese Schritte werden iterativ wiederholt. Am Schluss wird aus dem erhaltenen Consensus-Set die perspektivische Transformation berechnet. In OpenCv wird diese dann ebenfalls mit der Levenberg-Marquardt-Methode verfeinert. Da die RANSAC-basierte Methode robust gegen gröbere Ausreißer ist und einen definierbaren Schwellwert besitzt, kommt sie in diesem Projekt zur Anwendung.



2.3.3. Least-Median robuste Methode

LMedS (*Least-Median-Of-Squares*) [Peter J. Rousseeuw 1984] braucht im Gegensatz zu RANSAC keinen vordefinierten Schwellwert. Allerdings funktioniert die Methode nur, wenn über 50 Prozent der Punkte keine Ausreißer sind. LMedS arbeitet auf zufällig ausgewählten Untermengen. Die Methode funktioniert ähnlich wie die *Least-Square*-Methode, allerdings wird hier versucht, statt der Summe, den Median der kleinsten Quadrate zu minimieren. Da diese Methode mit unter 50 Prozent an Ausreißern zurecht kommt, zählt sie zu den robusten Methoden. Auch hier wird in OpenCV anschließend mit der Levenberg-Marquardt-Methode optimiert.

2.4. Der Canny-Algorithmus

Der Canny-Algorithmus [Canny 1986] ist ein Methode zur Kantenerkennung in Bildern. Entwickelt wurde er 1986 von John F. Canny, einem australischen Informatiker. Die Methode durchläuft einen mehrstufigen Prozess:

2.4.1. Glättung:

Zur Vorbereitung auf den eigentlichen Algorithmus wird das Bild in Graustufen konvertiert. Das Graustufen-Bild wird dann mittels Gaußverteilung geglättet, um unerwünschte Kanten, die durch Bildrauschen entstehen können, zu vermeiden.

2.4.2. Berechnen der partiellen Ableitungen:

Der Algorithmus nutzt vier Filter zur partiellen Ableitung, um horizontale, vertikale und diagonale Kanten zu berechnen. Dazu kann z.B der Sobel-Operator genutzt werden. Mithilfe der erhaltenen partiellen Ableitungen wird nun die Orientierung einer Kante mit dem Arkustangens berechnet. Der Wert wird auf 0° , 45° , 90° und 135° gerundet, entsprechend den acht Nachbarn eines Pixels. Danach wird die Kantenstärke bestimmt, indem man den euklidischen Betrag der partiellen Ableitungen bildet.

2.4.3. Non-maximum Supression:

Diese Methode hat das Ziel die Kanten dünner zu machen. Dabei werden die Kantenstärken benachbarter Pixel verglichen, um ein lokales Maximum zu finden. Der Pixel mit der maximalen Kantenstärke bleibt erhalten, der Rest wird auf Null gesetzt. So werden Kanten erzeugt, die nur einen Pixel breit sind.



2.4.4. Anwendung der Schwellwerte:

Da noch immer ungewollte Kantenfragmente durch Rauschen entstanden sein könnten, werden zwei Schwellwerte angewandt, um Pixel gewisser Kantenstärke zu verworfen. Der erste Schwellwert gibt an, ab welcher Kantenstärke ein Pixel als Kante angesehen werden soll. Der zweite Schwellwert bestimmt, welche Pixel zu der gefundenen Kante angerechnet werden.

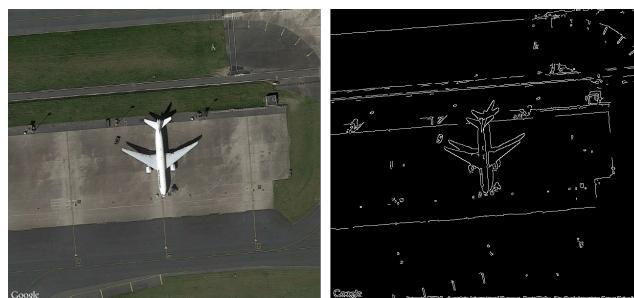


Abbildung 2.3.: Vor und nach Anwendung des Canny-Algorithmus.



3. Vorbereitung

Mit Hilfe der Google-Maps-API lassen sich Satellitenbilder an gewünschten Positionen mit unterschiedlichen Größen und Zoom-Stufen generieren. Dazu können gewünschte Parameter in einer URL-Anfrage spezifiziert werden. Der Parameter *center* gibt die Bildmitte an, *zoom* die Zoom-Stufe, *size* die Bildgröße, *format* das Format der ausgegebenen Datei, *maptype* die Art der Kartenansicht und *sensor* ob GPS-Daten übertragen werden sollen. Es sind noch weitere Parameter definierbar, doch sind diese, für die Zwecke dieses Projektes, nicht relevant.

```
String imageUrl = "http://maps.googleapis.com/maps/api/staticmap?" +
    "center=49.015384,2.510891" +
    "&zoom=18" +
    "&size=700x200" +
    "&format=png32" +
    "&maptype=satellite" +
    "&sensor=false";
```

Zur Vorbereitung wurden 16 beispielhafte Bilder in drei verschiedenen Zoom-Stufen erstellt. Sie enthalten ein oder mehrere Flugzeuge, teils in der Luft, teils am Boden stehend. Die folgenden Zoom-Stufen wurden verwendet: 16, 17, 18. Auf Stufen kleiner als 16 wurde verzichtet, da dann das zu erkennende Objekt aus zu wenig Pixeln für eine erfolgversprechende Auswertung besteht. Stufen größer als 18 wurden ebenfalls nicht erstellt, da das Objekt dann die gesamte Szene ausfüllen würde.

3.1. Das Objekt

Aus einem der Satellitenbilder wurde das zu erkennende Objekt mit Hilfe von Gimp extrahiert. Dabei wurde ein möglichst repräsentatives und klares Bild eines Flugzeuges gewählt. Dieses parkt auf einem Flugplatz und hat, für die Objekterkennung, keine schlechten Merkmale wie ungünstigen Schattenwurf, Bewegungsunschärfe oder Ähnliches. Es wurde ein Flugzeug mit zwei Triebwerken gewählt, da dies der Mehrheit der gefundenen Flugzeuge entspricht. Es wurden je drei Versionen des Objektes ausgeschnitten, entsprechend der drei verwendeten Zoom-Stufen. Es ergaben sich folgende Bildgrößen:



- Zoom 16: 45x43 Pixel
- Zoom 17: 83x88 Pixel
- Zoom 18: 165x172 Pixel

Da der Hintergrund die SURF-Merkmale des Objektes beeinflusst, wurden drei verschiedene Hintergrundfarben gewählt: Schwarz, Weiß und Grau.

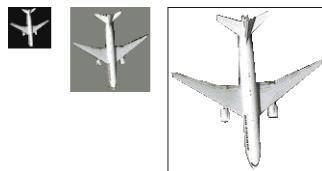


Abbildung 3.1.: Objekt in allen 3 Zoom-Stufen und Hintergrundfarben.

3.2. Die Szene

Das Bild, in dem das Objekt gesucht wird, wird im Folgenden Szene genannt. Es wurden 16 Szenen für dieses Projekt erstellt. Jede davon wurde jeweils in den drei verwendeten Zoom-Stufen generiert. Alle Bilder wurden in der, von Google-Maps beschränkten, maximalen Größe von 640x600 Pixeln erstellt.

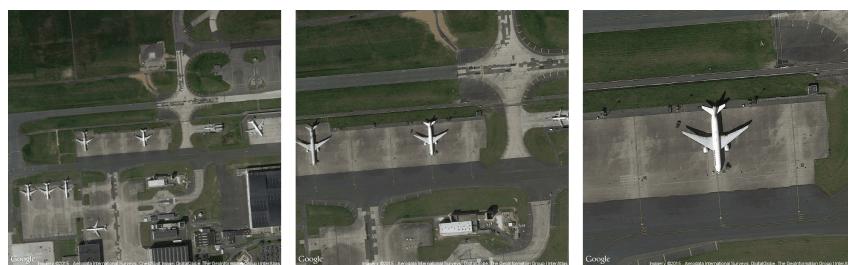


Abbildung 3.2.: Alle drei Zoom-Stufen einer Szene.

Hierbei ist anzumerken, dass, trotz selber Zoom-Stufe, die Größe der Flugzeuge in manchen Fällen stark variiert. Dies liegt an unterschiedlichen Flugzeugtypen und einer leicht unterschiedlichen Flughöhe.

Die 16 Szenen dienen zum testen des Algorithmus und zur Optimierung der Wahl der verschiedenen Parameter und Methoden. Um die Grenzen der Objekterkennung zu testen, wurden daher auch schwierige, undeutliche Beispiele gewählt.



Abbildung 3.3.: Zwei Flugzeuge unterschiedlicher Größe bei gleicher Zoom-Stufe.



Abbildung 3.4.: Beispiele für undeutliche, schwer zu erkennende Objekte.

3.3. Graphische Benutzeroberfläche

Um die Auswirkungen verschiedener Schwellwerte zu visualisieren, wurde eine graphische Benutzeroberfläche erstellt. Hier ist es möglich Objekte verschiedener Zoom-Stufen mit unterschiedlichen Hintergrundfarben zu wählen. Alle 16 Szenen in den verwendeten drei Zoom-Stufen sind auswählbar. Die durch den SURF-Algorithmus erzeugten Merkmale im Objekt und in der Szene werden durch farbige Linien verbunden. Dabei skaliert die Farbe der Linie von Blau zu Rot, je nach Größe der Differenz der beiden verbundenen Merkmale. So lässt sich schnell die Qualität eines Merkmals erkennen. Die besonders guten Merkmale, die einen Treffer anzeigen, werden in grün gezeichnet. Außerdem markiert ein grüner Rahmen ein gefundenes Objekt. Des Weiteren lassen sich folgende Parameter über die Benutzeroberfläche steuern:

- Auswahl, ob Canny- oder Graustufen-Bild verwendet werden soll
- Auswahl, ob Original oder bearbeitetes Bild angezeigt werden soll
- Auswahl, ob Treffer angezeigt werden soll
- Schieberegler für den Maximal-Abstand der Merkmals-Paare in der Ansicht
- Schieberegler für den Maximal-Abstand der Merkmals-Paare, die für die Objekterkennung genutzt werden
- Schieberegler für die beiden Schwellwerte des Canny-Algorithmus



3. Vorbereitung

- Schieberegler für den RANSAC-Schwellwert

Die graphische Benutzeroberfläche wurde nach dem klassischen *Model-View-Controller*-Prinzip programmiert. Das heißt, die Ansicht der Benutzeroberfläche (*view*) ist vom eigentlichen Modell (*model*) komplett getrennt und damit austauschbar. Die Parameterübergabe in beide Richtungen wird dabei von einer kontrollierenden Klasse (*controller*) geregelt.

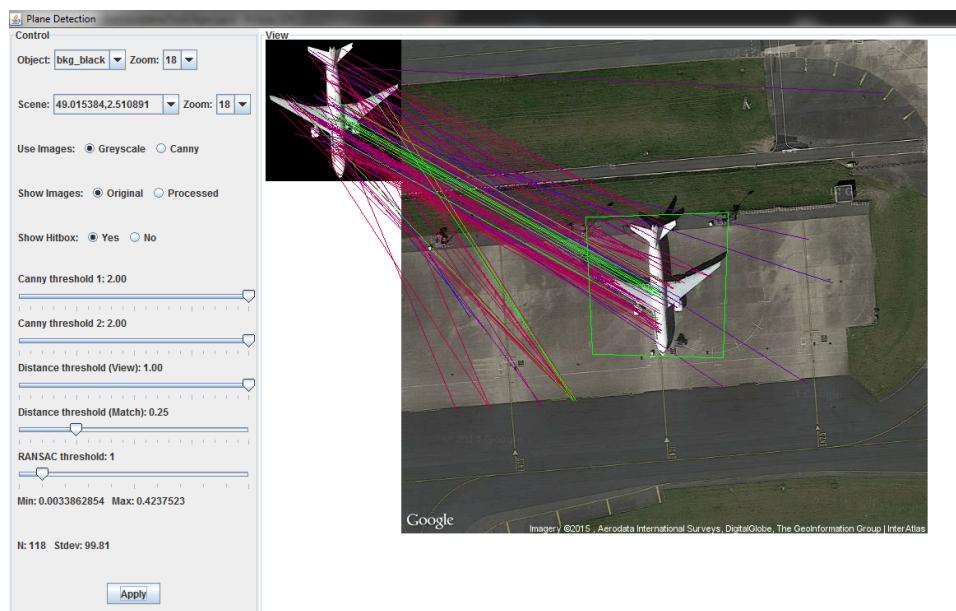


Abbildung 3.5.: Die graphische Benutzeroberfläche.



4. Anwendung

4.1. Verwendete Software

Das Projekt ist in Java implementiert. Als Entwicklungsumgebung diente Eclipse. Die Algorithmen zur Objekterkennung und Bildbearbeitung stammen aus der OpenCv-2.4.9 Bibliothek. Die Satellitenbilder wurden mithilfe der Google-Maps-API erstellt. Als Bildbearbeitungsprogramm diente GIMP.

4.2. Ablauf

Die Objekterkennung läuft in folgenden Schritten ab:

4.2.1. Erstellen der Graustufenbilder

Da der SURF und der Canny-Algorithmus nur auf Graustufenbildern arbeiten, werden die erstellten png-Dateien des Objekts und der Szene zu erst zu Graustufenbildern umgewandelt.

```
img_object = Highgui.imread(object, 0); //0 = CV_LOAD_IMAGE_GRAYSCALE
img_scene = Highgui.imread(scene, 0);
```

4.2.2. Anwendung des Canny-Algorithmus

Die Anwendung des Canny-Algorithmus ist optional und hängt davon ab, ob sie in der graphischen Benutzeroberfläche gewählt wurde. Der Canny-Algorithmus wird auf die Graustufenbilder des Objekts und der Szene angewandt und gibt Binärbilder der entsprechenden Graustufenbilder zurück. Auf diesen erscheinen dann die Kanten in weiß und der Hintergrund ist schwarz. Der Algorithmus nimmt zwei Schwellwerte als Argumente. Diese ergeben sich aus der Multiplikation des durchschnittlichen Grauwertes des jeweiligen Bildes mit den, in der Benutzeroberfläche eingestellten, Faktoren.



```
Imgproc.Canny(grey_object, img_object, cannytrsh1*meanGrey, cannytrsh2*meanGrey);
Imgproc.Canny(grey_scene, img_scene, cannytrsh1*meanGrey, cannytrsh2*meanGrey);
```

4.2.3. Anwendung des SURF-Algorithmus

Die Anwendung des SURF-Algorithmus erfolgt in zwei Stufen. In der ersten Stufen werden mittels einem *FeatureDetector* markante Merkmale aus Objekt und Szene ausgelesen und in zwei *KeyPoint*-Listen abgelegt.

```
FeatureDetector detector = FeatureDetector.create(FeatureDetector.SURF); //4 = SURF
detector.detect(img_object, keypoints_object);
detector.detect(img_scene, keypoints_scene);
```

In der zweiten Stufe extrahiert ein *DescriptorExtractor* die Deskriptoren der entsprechenden Merkmale aus den Bildern und fügt sie den *KeyPoints* in den Listen hinzu. Das Ergebnis ist eine Liste von SURF-Merkmalen für das Objekt und eine für die Szene.

```
DescriptorExtractor extractor = DescriptorExtractor.create(2); //2 = SURF;
extractor.compute(img_object, keypoints_object, descriptor_object);
extractor.compute(img_scene, keypoints_scene, descriptor_scene);
```

4.2.4. Bildung der Merkmals-Paare

Nun wird für jedes Objekt-Merkmal das Szene-Merkmal mit der geringsten Deskriptor-Differenz gesucht. Dafür wird ein *DescriptorMatcher* verwendet. Dieser gibt eine Liste aus Merkmals-Paaren zurück. Da diese Objekterkennung keine zeitkritische Anwendung ist, wird hier die Bruteforce-Methode verwendet.

```
DescriptorMatcher matcher = DescriptorMatcher.create(DescriptorMatcher.BRUTEFORCE);
matcher.match(descriptor_object, descriptor_scene, matches);
```

4.2.5. Schwellwert-Filter

Um den, von möglichen Ausreißern erzeugten, Fehler gering zu halten, werden nur die Merkmals-Paare beibehalten, deren Deskriptor-Distanz einen, in der Benutzeroberfläche definierten, Schwellwert nicht überschreiten. Dieser Schwellwert wird gebildet aus der Differenz zwischen maximaler und minimaler Distanz und einem definierten Schwellwert-Faktor.



```
float thresh = this.min + matchThresh*(this.max - this.min);

for(int i = 0; i < this.kpMatches.size();i++){
    if (this.kpMatches.get(i).getDist()< thresh){
        goodMatches.add(this.kpMatches.get(i));
    }
}
```

4.2.6. Perspektivische Transformation

Um die Position und Orientierung eines Treffers in der Szene zu berechnen, wird die perspektivische Transformation gesucht, die die Objekt-Merkmale auf die Szenen-Merkmale abbildet. Es wurde die RANSAC-Methode gewählt, da diese robust auf Ausreißer reagiert. Der RANSAC-Schwellwert wird über die Benutzeroberfläche definiert.

```
Mat hg = Calib3d.findHomography(obj, scene, Calib3d.RANSAC, ransacThresh);
```

4.2.7. Anwendung der perspektivischen Transformation

Ist die perspektivische Transformation gefunden, kann sie auf die Koordinaten der Objekt-Merkmale aus der Liste der Merkmals-Paare angewendet werden. Man erhält so die Position und die Orientierung eines Treffers in der Szene.

```
Core.perspectiveTransform(obj_corners,scene_corners, hg);
```



5. Auswertung

5.1. Parameterwahl

Mithilfe der graphischen Benutzeroberfläche und den 16 verschiedenen Szenen konnte die Auswirkung der einstellbaren Parameter auf die Qualität der Objekterkennung beobachtet werden. Das Ziel dabei war es, diejenige Parametereinstellung zu finden, die ein optimales Ergebnis für alle 16 Szenen gewährleistet. Mit dieser optimalen Parameterwahl könnte dann eine großflächige Suche nach Flugzeugen in weiteren, beliebigen Satellitenbildern gestartet werden.

Begonnen wurde damit, die optimale Einstellung für die Szene zu finden, aus der das Objekt ausgeschnitten wurde. Danach wurde diese Einstellung auf den anderen 15 Szenen getestet und gegebenenfalls verändert. Folgende Parametereinstellung konnte mithilfe der graphischen Benutzeroberfläche ermittelt werden:

5.1.1. Zoom-Stufe

Es stehen drei Zoom-Stufen zur Auswahl: 16, 17 und 18. Je größer die Zoom-Stufe desto mehr Details lassen sich erkennen. Die Zoom-Stufe hat maßgeblich Einfluss auf die Größe des Objektes und damit auf die Anzahl der extrahierten SURF-Merkmale. Die Anzahl der Merkmale auf den verschiedenen Zoom-Stufen ergibt sich wie folgt:

Zoom-Stufe	Hintergrundfarbe	Merkmals-Anzahl
16	Weiß	3
	Grau	7
	Schwarz	5
17	Weiß	51
	Grau	32
	Schwarz	33
18	Weiß	150
	Grau	119
	Schwarz	118

Tabelle 5.1.: Anzahl der extrahierten Merkmale aus den Graustufenbildern mit unterschiedlichen Zoom-Stufen und Hintergrundfarben.



5. Auswertung

Da für die Funktion *findHomography*, zur Identifikation eines Treffers, mindestens vier Merkmals-Paare benötigt werden, ist die Zoom-Stufe 16 mit maximal 5 möglichen Merkmals-Paaren komplett ungeeignet.

Zoom-Stufe 17 erwies sich ebenfalls als ungeeignet. Es war nicht möglich das Objekt in der Szene, aus der es extrahiert wurde, zu identifizieren. Die Merkmals-Paare streuten zu stark und hatten eine zu hohe Differenz.

Auf Zoom-Stufe 18 konnte das Objekt in der Szene, aus der es extrahiert wurde, identifiziert werden. Man erkennt auf dieser Stufe kleine Details, wie z.B ein Logo auf der Seite des Rumpfes, die in den anderen Zoom-Stufen nicht erkennbar sind. Diese Details innerhalb des Flugzeugkörpers werden in der Szene wiedererkannt und liefern gute Merkmals-Paare. Zoom-Stufe 18 ist damit Teil der optimalen Parameterwahl. Des Weiteren wurden Testfälle untersucht, bei denen die Zoom-Stufe des Objekts und die Zoom-Stufe der Szene unterschiedlich waren. Dies führte jedoch immer zu schlechten Ergebnissen.

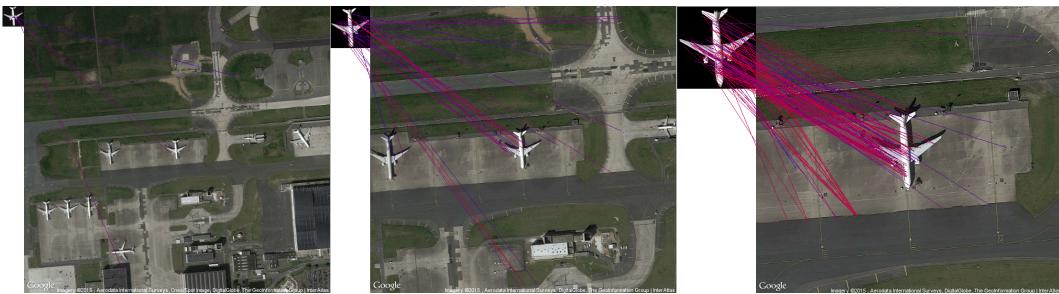


Abbildung 5.1.: Zoom-Stufe 16, 17 und 18.

5.1.2. Hintergrundfarbe des Objekts

Für den Hintergrund des Objektes wurden drei verschiedene Graustufen gewählt: Schwarz, Weiß und Grau. Auf farbige Hintergründe wurde verzichtet, da der implementierte SURF-Algorithmus nur mit Graustufenbildern arbeitet.

Hintergrundfarbe	N	Min	Max	Stdev
Weiß	150	0.018	0.531	178.9
Grau	119	0.008	0.551	146.9
Schwarz	118	0.003	0.424	99.8

Tabelle 5.2.: Statistik der Merkmals-Paare am Beispiel der Szene, aus der das gesuchte Objekt stammt.

Die Hintergrundfarbe Weiß lieferte die schlechtesten Ergebnisse. Die Merkmals-Paare streuten sehr stark und hatten eine sehr hohe Differenz. Es war damit nicht



5. Auswertung

möglich, das Objekt in seiner Ursprungs-Szene zu erkennen.

Die Hintergrundfarbe Grau lieferte bessere Ergebnisse. Das Objekt konnte, trotz Streuung der Merkmals-Paare, in seiner Ursprungs-Szene identifiziert werden.

Die Hintergrundfarbe Schwarz lieferte die besten Ergebnisse. Das Objekt konnte ebenfalls identifiziert werden. Allerdings war der Abstand der Merkmals-Paare deutlich geringer und die Zuordnung der Treffer war deutlicher zu erkennen. Außerdem ließen sich auch in anderen Szenen Treffer erzielen. Somit ist Schwarz Teil der optimalen Parameterwahl.

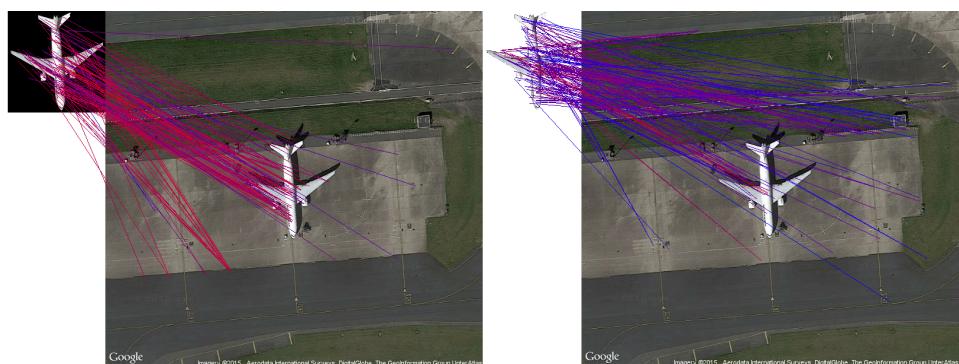


Abbildung 5.2.: Beispiel für die Auswirkung unterschiedlicher Objekt-Hintergründe (Schwarz und Weiß) auf die Qualität der Merkmals-Paare.

5.1.3. Canny Schwellwerte

Zuerst wurde der Canny-Algorithmus mit absoluten Schwellwerten getestet. Dies lieferte jedoch keine guten Ergebnisse. Die Objekterkennung in einer bestimmten Szene konnte, durch Veränderung der Schwellwerte, individuell verbessert werden. Da jede Szene eine eigene Grund-Helligkeit besitzt, wurde zu relativen Schwellwerten gewechselt. Die Schwellwerte können als Prozent des durchschnittlichen Grauwertes einer Szene angegeben werden. Die besten Resultate wurden erzielt mit einem unteren Schwellwert von 1.0 und einem oberen Schwellwert von 1.6.

5.1.4. Maximale Distanz der Merkmals-Paare

Die maximale Distanz der Merkmals-Paare gibt an, welche Merkmals-Paare zur Berechnung eines Treffers beitragen. Ist die Distanz zweier Merkmale zu groß, werden sie bei der Treffer-Identifikation nicht berücksichtigt. Da jede Szene eine individuelle Spanne von Merkmals-Paar-Distanzen besitzt, ist ein absoluter Schwellwert ungeeignet. Der Schwellwert wurde deshalb relativ zur maximalen Ausdehnung der



5. Auswertung

Merkmals-Paar-Distanzen gewählt. Der Wert für *matchThresh* ist hier der einstellbare Parameter:

```
float thresh = this.min + matchThresh*(this.max - this.min);
```

Die besten Ergebnisse lassen sich mit einem Schwellwert zwischen 0.35 und 0.4 erzielen. Allerdings erwies sich der RANSAC-Algorithmus bei vielen Ausreißern als sehr robust. Tatsächlich konnte teilweise eine besser definierte Hitbox gefunden werden, indem man alle Merkmals-Paare zur Berechnung berücksichtigte und dem RANSAC-Algorithmus die Filterung der Ausreißer überließ.

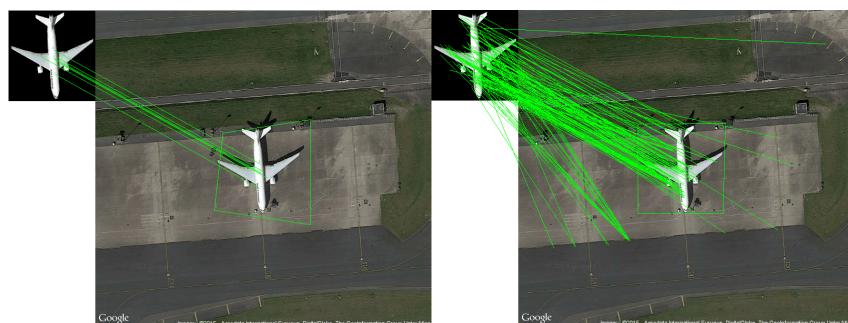


Abbildung 5.3.: Beispiel: Die Berücksichtigung aller Merkmals-Paare ergibt eine besser definierte Hitbox.

5.1.5. RANSAC Schwellwert

Dieser Schwellwert bestimmt, ob ein Merkmals-Paar zur Berechnung der Treffer-Hitbox genutzt wird, oder ob es, als Ausreißer, nicht berücksichtigt wird. Ist der Schwellwert zu hoch wird die Hitbox durch weit entfernte oder ungünstig liegende Punkte verfälscht. Ist der Schwellwert zu niedrig, werden möglicherweise relevante Punkte bei der Berechnung nicht berücksichtigt.



Abbildung 5.4.: Beispiel für eine perspektivische Abbildung, die die Hitbox verzerrt.



Es lassen sich Schwellwerte zwischen 1 und 10 angeben. Die besten Ergebnisse ergab ein Wert von 3. Dabei ist die errechnete Hitbox nicht immer rechteckig. Teilweise ergibt sich ein viereckiges Polygon, welches das Objekt aber umschließt.

5.1.6. Graustufen oder Canny

Mit einem Graustufen-Bild als Input für den SURF-Algorithmus war es möglich insgesamt 2 von 16 Flugzeugen zu erkennen. Aufgrund dieses schlechten Ergebnisses wurde das Graustufen-Bild mit dem Canny-Algorithmus bearbeitet. Das Canny-Bild als Input für den SURF-Algorithmus lieferte allerdings noch schlechtere Ergebnisse. Außer in der Szene, aus der das Objekt ursprünglich extrahiert wurde, konnte kein weiterer Treffer identifiziert werden.

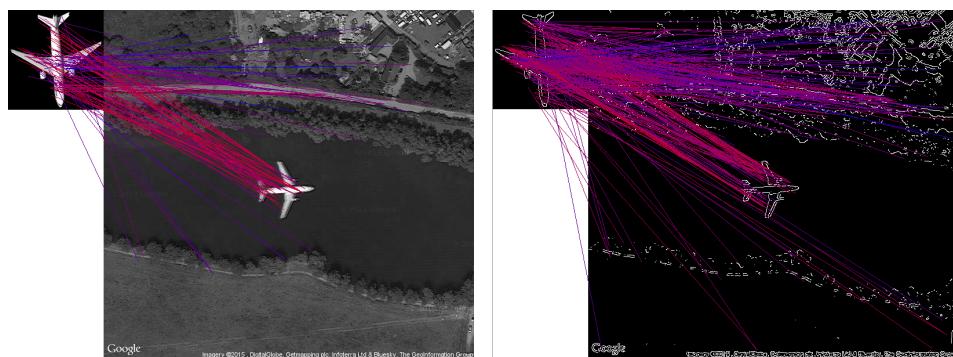


Abbildung 5.5.: Vergleich zwischen Graustufen- und Canny-Bild.

5.1.7. Zusammenfassung

Als optimale Parameter ergaben sich:

- Zoom-Stufe: 18
- Hintergrundfarbe: Schwarz
- Canny-Schwellwerte: 1.0 und 1.6
- Maximal-Distanz-Schwellwert: 0.35-0.4
- RANSAC Schwellwert: 3
- Graustufenbild



6. Fazit und kritische Bewertung

Es konnten nur zwei Flugzeuge in den insgesamt 16 Szenen gefunden werden. Dieses schlechte Ergebnis kann nicht auf eine suboptimale Wahl der Parameter zurückgeführt werden. Es liegt vielmehr in der Natur der Objekte und der Art der Merkmals-Extraktion des SURF-Algorithmus. Ein Flugzeug wird vor allem durch seine Form und durch spezifische Proportionen zwischen Rumpf und Tragfläche definiert. Der SURF-Algorithmus sucht nach markanten Merkmalen in verschiedenen Skalierungsebenen. Dabei geben nur die großflächigen Merkmale auf höchster Ebene diese, für ein Flugzeug, spezifischen Merkmale der Form wieder. Merkmale auf niedrigeren Ebenen beziehen sich auf Details die am Flugzeugrumpf gefunden werden, wie z.B. ein Logo oder Antennenvorrichtungen. Diese Merkmale unterscheiden sich aber bei verschiedenen Flugzeugtypen und sind daher nicht geeignet für die Objekterkennung. Ein weiterer ungünstiger Faktor ist der Hintergrund des Objektes. Da die SURF-Merkmale in dem gesamten rechteckigen Bild gesucht werden, entstehen Merkmale, die stark vom Hintergrund beeinflusst sind. Da der Hintergrund bei gesuchten Objekten aber ständig wechselt, sind diese Merkmale ebenfalls ungeeignet. Auch wurden sehr allgemeine Merkmale durch den schwarzen Hintergrund und den Kontrast zum weißen Flugzeug erzeugt, die in einer Szene auch oft in anderen Objekten, wie, z.B. bei einer dunklen Straße auf hellem Hintergrund, auftreten. Somit lässt sich abschließend sagen, dass die Anwendung von SURF-Merkmalen zur Flugzeugerkennung in Satellitenbildern nicht geeignet ist. Aufgrund dieses schlechten Ergebnisses wurde auf eine großflächige Suche nach Flugzeugen in beliebigen Satellitenbildern verzichtet.



Literaturverzeichnis

Bay u. a. 2006

BAY, Herbert ; TUYTELAARS, Tinne ; GOOL, Luc V.: SURF: Speeded Up Robust Features. In: *Proceedings of the ninth European Conference on Computer Vision*, 2006 [2.1](#)

Bay 2008

BAY, Herbert et al.: SURF: Speeded Up Robust Features. In: *Computer Vision and Image Understanding (CVIU)* (2008) [2.2](#)

Canny 1986

CANNY, J: A Computational Approach to Edge Detection. In: *IEEE Trans. Pattern Anal. Mach. Intell.* 8 (1986), Juni, Nr. 6, S. 679–698. – ISSN 0162-8828 [2.4](#)

Fischler und Bolles 1981

FISCHLER, Martin A. ; BOLLES, Robert C.: Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. In: *Communications of the ACM* 24 (1981), Nr. 6, S. 381–395 [2.3.2](#)

Haar 1910

HAAR, A.: Zur Theorie der Orthogonalen Funktionensysteme. (German) [On the theory of orthogonal function systems]. 69 (1910), S. 331–371. – ISSN 0025–5831 (print), 1432–1807 (electronic) [2.1.2](#)

Muja und Lowe 2014

MUJA, Marius ; LOWE, David G.: Scalable Nearest Neighbor Algorithms for High Dimensional Data. In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 36 (2014) [2.2.2](#)

Peter J. Rousseeuw 1984

PETER J. ROUSSEEUW: Least Median of Squares Regression. In: *Journal of the American Statistical Association* 79 (1984), December, Nr. 388, S. 871, , 880 [2.3.3](#)

Viola und Jones 2001

VIOLA, Paul ; JONES, Michael: Rapid object detection using a boosted cascade of simple features, 2001, S. 511–518 [2.1.1](#)



A. Anhang

A.1. Eidesstattliche Erklärung

Ich, Felice Serra, Matrikel-Nr. 2944562, versichere hiermit, dass ich meine Studienarbeit mit dem Thema

Anwendung des SURF-Algorithmus Flugzeugerkennung in Satellitenbildern

selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe, wobei ich alle wörtlichen und sinngemäßen Zitate als solche gekennzeichnet habe. Die Arbeit wurde bisher keiner anderen Prüfungsbehörde vorgelegt und auch nicht veröffentlicht.

Tübingen, den 9. März 2015

FELICE SERRA

A.2. Adresse

Felice Serra
Schlachthausstraße 30
72074 Tübingen
E-Mail: felice.serra@gmx.de
Tel.: 0178-8017472