UNIVERSITÄT
KOBLENZ · LANDAU
Fachbereich 4: Informatik

PokAR: Spielkarten als Marker

Studienarbeit

im Studiengang Computervisualistik

vorgelegt von
Lubosz Podeszwa

Betreuer: Prof. Dr. Stefan Müller
(Institut für Computervisualistik, AG Computergraphik)

Koblenz, im Januar 2010

Erklärung

Ich versichere, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

	Ja	Nein
Mit der Einstellung der Arbeit in die Bibliothek bin ich einverstanden.	☐	☐
Der Veröffentlichung dieser Arbeit im Internet stimme ich zu.	☐	☐

.....
(Ort, Datum) (Unterschrift)

Inhaltsverzeichnis

1 Einleitung

1.1 Motivation

Pokern erfreut sich einer immer größer werdenden Beliebtheit. Seitdem das Pokern über Internet immer populärer wird, haben viele User Ihre Leidenschaft für das Glücksspiel neu entdeckt. Besonders beliebt ist dabei die Variante *Texas Hold'Em*, in der es im Vergleich zum klassischen Poker neben den Karten auf der Hand auch noch eine gewisse Anzahl von Gemeinschaftskarten gibt, die nach und nach aufgedeckt werden. Dadurch erhöht sich der strategische Anteil enorm. Jeder Spieler kennt außer seinen eigenen Karten auch einen Teil der gegnerischen Karten und muss überlegen, welche Wendung die nächste aufgedeckte Gemeinschaftskarte herbeiführen kann.
Dabei gibt es unzählige Strategien, die dem Spieler Verhaltensweisen zu seinen Karten näher bringen. Diese beruhen auf komplexen mathematischen Berechnungen und gewährleisten beim Einhalten bestimmter Regeln eine statistisch sehr hohe Gewinnrate. Doch das Regelwerk bei diesen Strategien ist ziemlich groß. Es ist genau festgelegt wie sich der Spieler verhalten soll. Die Strategie fordert penible Einhaltung aller Regeln, damit sie zum Erfolg führt. Dadurch wird aus dem Glücksspiel Poker ein mathematisch vorbestimmter Handlungsablauf, der keinen Platz zum bluffen lässt. Außerdem lernt der Spieler auf diese Art nicht viel über die Wahrscheinlichkeiten, die das Pokerspiel birgt, da er alle Handlungsweisen in seinem Strategieregelwerk nachschlagen kann und nicht selber denken muss.

Welche Möglichkeiten bleiben also noch um seine Gewinnchancen oder die Chancen auf eine gute Hand abschätzen zu können? Im Bereich des Internet-Pokerns hat der Nutzer die Möglichkeit auf bestimmte Programme mit manueller Eingabe der aktuellen Spielkarten oder auf sogenannte *Pokerbots* zurückzugreifen. Diese Pokerbots lesen aus dem Speicher die Karten vom Bildschirm aus und sind so in der Lage dem Spieler entweder durch Angabe der Wahrscheinlichkeiten zur Seite zu stehen oder sogar die Steuerung des Spiels für den User zu übernehmen.

Doch wie kann ein Spieler ohne mathematische Kenntnisse oder langjährige Pokererfahrung in einem realen Pokerspiel etwas über die Wahrscheinlichkeiten seiner Pokerhand erfahren? Wie wahrscheinlich ist es, dass er mit dem nächsten Blatt eine Straße oder ein Full House bekommt?

1.2 Zielsetzung

Ziel dieser Studienarbeit ist es, eine Applikation zu entwickeln, die dem User aufgrund von Spielkarten aus einem Pokerdeck Wahrscheinlichkeiten für den Weitern Spielverlauf berechnet. Die Spielkarten werden dabei über eine gewöhnliche Webcam eingelesen und erkannt. Die Applikation soll invariant gegenüber verschiedenen Skalierungen und Ausrichtungen der Karten, sowie leichten Beleuchtungsunterschieden sein. Dabei sollen die Karten selbst als Marker erkannt werden. Entsprechend dem Texas Hold'Em Regelwerk sollen dem User zu verschiedenen Phasen des Spiels aktuelle hilfreiche Informationen geliefert werden. Folgende Informationen sollen darunter enthalten sein:

- Wahrscheinlichkeit in den nächsten Spielrunden mit den noch nicht aufgedeckten Karten eine bestimmte Hand zu bekommen
- Die beste Hand die sich aus den aktuelle Karten bilden lässt
- Die beste Hand nach Beendigung der letzten Spielphase
- Wahrscheinlichkeit eine bessere Hand zu haben als die Gegenspieler

Mithilfe dieser Applikation soll es Pokeranfängern möglich sein, ihre eigene Hand besser einzuschätzen und dementsprechend zu agieren. Dabei sollen dem Spieler aufgrund der Spielkarten oder der Wahrscheinlichkeiten keine Spielzüge suggeriert werde. Es sollen lediglich die mathematischen Wahrscheinlichkeiten auf potentielle Hände vermittelt werden, so dass der Spieler aufgrund dieser Informationen selbst über den weiteren Verlauf des Spiels bestimmen kann.

1.3 Aufbau

Im Kapitel 2 soll in die Grundlagen von Texas Hold'em eingeführt werden. Es wird ein Überblick gegeben über die Spielregeln, den Ablauf einer Pokerpartie und die Wahrscheinlichkeiten die berechnet werden können. Kapitel 3 befasst sich mit den Grundlangen der Objekterkennung. Es werden verschiedene Methoden wie der Canny Kantendetektor, die Hough-Transformation oder das Template Matching vorgestellt. Im Kapitel 4 wird Tracking vorgestellt. Dabei wird auf markerbasiertes und markerloses Tracking eingegangen. Die genaue Implementation der Studienarbeit wird im Kapitel 5 beschrieben. In den Kapiteln 6 um 7 werden die Ergebnisse der Studienarbeit und die Ausblicke auf mögliche Erweiterungen vorgestellt.

2 Texas Hold'Em

2.1 Spielprinzip

Diese Studienarbeit basiert auf der Pokervariante Texas Hold'em. Bei dieser Variante setzen sich die Hände der Spieler aus zwei Handkarten, die nur die Spieler selbst einsehen können, und aus bis zu fünf Gemeinschaftskarten, die von jedem Spieler eingesehen und benutzt werden können, zusammen. Eine Hand kann dabei beliebig aus diesen Handkarten und Gemeinschaftskarten zusammengestellt werden, muss aber genau fünf Karten umfassen. Dabei ist es egal, ob der Spieler beide, eine oder keine seiner Handkarten in seine Pokerhand mit einbringt. Jede Runde besteht aus mehreren Phasen, in denen abwechselnd Karten verteilt bzw. aufgedeckt werden und gesetzt werden können. Die einzelnen Phasen jeder Spielrunde werden im nächsten Kapitel genauer erläutert.

2.2 Ablauf

Jede Pokerrunde besteht aus fünf Phasen. In den ersten vier Phasen (Preflop, Flop, Turn, River) jeweils eine gewisse Anzahl an Karten verteilt bzw. aufgedeckt. Nach jeder dieser Phasen folgt eine Bietrunde, in der alle Spieler ihre Wetteinsätze machen können. Die fünfte und letzte Phase jeder Pokerrunde ist der Showdown, bei dem alle noch verbleibenden Spieler ihre beiden Handkarten offen legen. Nun wird verglichen, welche Kombination aus Hand- und Gemeinschaftskarten den höchsten Handwert ergibt. Der genaue Ablauf der einzelnen Phasen sieht wie folgt aus:

- **Preflop** Jeder Spieler erhält zwei Handkarten, die nur er einsehen und in die finale Pokerhand mit einbringen kann
- **Bietrunde** Reihum hat nun jeder Spieler die Möglichkeit, solange es die Spielsituation zulässt, zu *checken* (nichts zu tun), zu *callen* (mitzugehen), zu *raisen* (den gebotenen Spieleinsatz zu erhöhen) oder zu *folden* (aus dem Spiel auszusteigen). Jede Bietrunde im folgenden Spielverlauf ist dabei gleich aufgebaut.
- **Flop** Es werden drei Karten aufgedeckt und offen in die Tischmitte gelegt, so dass jeder Spieler sie einsehen kann. Diese Karten gehören zu den sogenannten Gemeinschaftskarten.
- **Bietrunde**
- **Turn** Es wird eine weitere Karte aufgedeckt und den Gemeinschaftskarten hinzugefügt.
- **Bietrunde**

- **River** Die letzte Karte wird nun aufgedeckt und den Gemeinschaftskarten hinzugefügt. Nun wurden alle Karten für diese Spielrunde aufgedeckt.
- **Bietrunde**
- **Showdown** Alle Spieler, die noch im Spiel sind decken nun ihre Karten auf, aus denen der Gewinner dieser Runde ermittelt wird.

2.3 Hände

Eine Hand besteht im Pokern aus fünf Karten und wird nach der Höhe der Kombination bewertet. Je geringer die Wahrscheinlichkeit für eine Kombination ist, desto besser ist die Hand. Falls mehrere Spieler die gleiche Haben haben, entscheidet die Beikarte bzw. die Beikarten (auch Kicker genannt), welche Hand stärker ist. Als Beikarten werden Karten bezeichnet, die nicht zur Bestimmung des Handwertes beitragen, zum Beispiel besteht die Hand „Drilling“ aus drei Karten des selben Wertes und zwei Beikarten. Sollten mehrere Spieler allerdings dieselben fünf Karten haben, kommt es in der Regel zu einem Unentschieden, und der Pot wird unter diesen Spielern aufgeteilt (*Split Pot*). Die Farben spielen dabei keine Rolle.
In Abbildung **??** sind alle möglichen Hände veranschaulicht.

2.4 Wahrscheinlichkeiten

Wahrscheinlichkeiten spielen beim Pokern eine sehr große Rolle. Jedoch ist die mathematische Berechnung dieser Wahrscheinlichkeiten sehr komplex. Allein schon die Berechnung der Anzahl aller möglicher Hände gestaltet sich schwieriger als man vielleicht annehmen könnte. Dies soll nun verdeutlicht werden:
Es gibt $\binom{52}{5}$ = 2.598.960 Möglichkeiten verschiedene Hände aus einem Deck aus 52 Karten zu bekommen. Wenn man von der Variante Texas Hold'Em ausgeht, stehen einem Spieler sieben Karten von 52 zur Verfügung, D.h. es gibt $\binom{52}{7}$ = 133.784.560 verschiedene Kombinationen aus Karten, die der Spieler in der letzten Runde haben kann. Wenn in diesem Fall jetzt noch fünf aus diesen sieben Karten eine Hand bilden sollen, ist die Anzahl der Möglichkeiten $\binom{52}{5} \cdot \binom{47}{2}$ = 2.809.475.760. Rechnet man dazu noch die einzelnen Spielrunden ein ergeben sich $\binom{52}{2} \cdot \binom{50}{3} \cdot \binom{47}{1} \cdot \binom{46}{1}$ = 56.189.515.200 Möglichkeiten im kompletten Verlauf einer Runde.
Da sich schon die Bestimmung der Anzahl aller möglichen Hände schwierig gestaltet, lässt sich erahnen, dass die mathematische Beschreibung der Vorkommen bestimmter Hände relativ komplex ist. Beim normalen Pokern, bei dem der Spieler lediglich fünf Karten auf der Hand hat, ist die mathematische Beschreibung aufgrund der Eindeutigkeit der Hand noch relativ einfach.

Royal Flush Eine Folge von As, König, Dame, Bube und 10 in der gleichen Farbe *Beispiel*: TD, JD, QD, KD, AD

Straight Flush aufeinanderfolgende Karten in der gleichen Spielfarbe, z.B. 9 bis König. *Beispiel*: 5C, 6C, 7C, 8C, 9C

Four of a Kind Vier gleiche Karten unabhängig von der Farbe *Beispiel*: QH, QC, QD, QS

Full House Ein Drilling (z.B. 3 Könige) und ein Paar (z.B. zwei 8), unabhängig von Farbe *Beispiel*: JS, JH, 10S, 10C, 10D

Flush Fünf farbgleiche Karten, unabhängig von den Werten *Beispiel*: 4D, 6D, 9D, KD, AD

Straight 5 Werte in einer Folge, unabhängig von der Farbe *Beispiel*: 7, 8, 9, 10, J;

Three of a Kind 3 gleiche Karten unabhängig von der Farbe *Beispiel*: JH, JC, JD

Two Pair 2 Paare von 2 gleichen Karten *Beispiel*: 2S, 2C, 6H, 6S

One Pair 2 gleiche Karten, Farbe ist egal *Beispiel*: 10H, 10D

Highest Card Wenn niemand am Tisch mindestens ein Paar auf der Hand hat, gewinnt die höchste Karte (As, König...) *Beispiel*: KH

Abbildung 1: Liste aller Hände

Ein Straße lässt sich z.B. folgendermaßen definieren:
Eine Straße wird durch jede der zehn möglichen Sequenzen, die fünf aufeinander folgende Karten bilden können beschrieben (von A-2-3-4-5 bis 10-B-D-K-A]. Jede dieser fünf Karten kann eine beliebige der vier Farben enthalten. Außerdem müssen alle Straßen, die auch auch ein *Straight Flush* bilden ausgeschlossen werden. Mathematisch kann dies so beschrieben werden:
$\binom{10}{1}\binom{4}{1} - \binom{10}{1}\binom{4}{1} = 10,200$
Auf ähnliche Weise können alle Wahrscheinlichkeiten und Häufigkeiten für Pokerhände aus 5 Karten berechnet werde:

Hand	Häufigkeit	Wahrscheinlichkeit	Mathematische Beschreibung
Royal Flush	4	0.000154%	$\binom{4}{1}$
Straight Flush	36	0.00139%	$\binom{9}{1}\binom{4}{1}$
Vierling	624	0.0240%	$\binom{13}{1}\binom{12}{1}\binom{4}{1}$
Full House	7,744	0.144%	$\binom{13}{1}\binom{4}{3}\binom{12}{1}\binom{4}{2}$
Flush	5,108	0.197%	$\binom{13}{5}\binom{4}{1} - \binom{10}{1}\binom{4}{1}$
Strasse	10,200	0.392%	$\binom{10}{1}\binom{4}{1}^5 - \binom{10}{1}\binom{4}{1}$
Drilling	55,912	2.11%	$\binom{13}{1}\binom{4}{3}\binom{12}{2}\binom{4}{1}^2$
Zwei Paare	123,552	4.75%	$\binom{13}{2}\binom{4}{2}^2\binom{11}{1}\binom{4}{1}$
Paar	1,098,240	42.3%	$\binom{13}{1}\binom{4}{2}\binom{12}{3}\binom{4}{1}^3$
Höchste Karte	1,302,540	50.1%	$\left[\binom{13}{5} - 10\right]\left[\binom{4}{1}^5 - 4\right]$
Total	2,598,960	100%	$\binom{52}{5}$

Beim Texas Hold'Em lassen sich sieben Karten kombinieren. Versucht man hierbei nun ein solches mathematisches Regelwerk für die Wahrscheinlichkeiten und Häufigkeiten zu erstellen, wird man feststellen, dass durch die zwei zusätzlichen Karten unzählige Ausnahmen hinzukommen. Zum Beispiel könnten nun zu jedem Set aus sieben Karten mehrere unterschiedliche Kartenwerte ermittelt werden, von denen jedoch nur der höchste gültig wäre.
Diese und sehr viele andere Ausnahmeregelungen führen dazu, dass die mathematischen Beschreibungen extrem komplex werden. Eine Straße, die vorhin bei nur fünf Karten mit $\binom{10}{1}\binom{4}{1} - \binom{10}{1}\binom{4}{1} = 10,200$ beschrieben werden konnte, würde jetzt wie folgt aussehen:

$$\left[\binom{8}{2} + \binom{9}{1}\binom{7}{2}\right]\left[\binom{4}{1}^7 - \binom{4}{1}\left[1 + \binom{7}{6}\binom{3}{1} + \binom{7}{5}\binom{3}{1}^2\right]\right] +$$
$$\left[\binom{8}{1} + \binom{9}{1}\binom{7}{1}\right]\left[\binom{6}{1}\binom{4}{2}\right]\left[\binom{4}{2}^5 - \left[\binom{4}{1} + \binom{5}{4}\binom{2}{1}\binom{3}{1}\right]\right] + \binom{5}{1}\binom{5}{1}\binom{4}{3}\left[\binom{4}{1}^4 - \binom{3}{1}\right] +$$
$$\binom{5}{2}\binom{4}{2}^2\left[6 \cdot \left[64 - \binom{2}{1}\right] + 24 \cdot (64 - 1) + 6 \cdot 64\right] = 6,180,020$$

Aus dieser und den restlichen Regeln lassen sich nun die Wahrscheinlichkeiten und Häufigkeiten für eine aus sieben Karten zusammengestellte Hand errechnen:

Hand	Häufigkeit	Wahrscheinlichkeit
Royal Flush	4.324	0,003232%
Straight Flush	37.260	0,02785%
Vierling	224,848	0.168%
Full House	3,473,184	2.60%
Flush	4,047,644	3.03%
Strasse	6,180,020	4.62%
Drilling	6,461,620	4.83%
Zwei Paare	31,433,400	23.5%
Paar	58,627,800	43.8%
Höchste Karte	23,294,460	17.4%
Total	133,784,560	100%

3 Objekterkennung

In diesem Abschnitt sollen einige grundlegende Methoden und Ansätze der Objekterkennung kurz erklärt werden. Dazu sollte zuerst darauf eingegangen werden, wie der Aufbau eines Objekterkennungssystems aussehen kann.
Die einfachste Methode ein Objekt zu erkennen wäre das zu erkennende Objekt zu fotografieren und dieses Objektbild dann pixelweise mit einem unbekannten Bild, das das Objekt enthält, zu vergleichen. Da diese Methode allerdings nicht invariant ist gegenüber Skalierung und Rotation des Objekts, sowie gegenüber Veränderungen des Blickwinkels und der Belichtung, wäre sie sehr fehleranfällig. Außerdem würden, da das ganze Bild durchsucht wird, auch runde Objekte verglichen werden, obwohl eckige sucht. Um dies zu verhindern werden Informationsebenen gebildet, die das Bild in verschiedene Komplexitäsebenen einteilen.
Die unterste Ebene eines Objekterkennungssystems enthält die Sensordaten, also die rohen Bilddaten. Die nächste Ebene, die auf den Ergebnissen der vorherigen aufbaut, versucht nähere Informationen zu extrahieren, um die Komplexität des Informationsgehalts zu erhöhen. In der obersten Ebene, in der die Komplexität am höchsten ist, sind genug Informationen enthalten, um das gesuchte Objekt zu erkennen. Um Aussagen über das Vorhandensein eines Objektes im Ganzen machen zu können, muss sich ein Objekterkennungssystem also von der untersten Ebene nach oben hoch arbeiten.

Generell müssen folgende Schritte durchlaufen werden:

- Sensor wählen
- Vorverarbeitung der gewonnen Bilder zu Reduktion von Aufnahmestörungen
- Aufbereitetes Bild nach vorher festgelegten Merkmalen durchsuchen (z.B. Kanten, Ecken, Farbe oder Reflektanz)
- Extraktion der gefundenen Merkmale in den Attributraum (bzw. Merkmals- oder Eigenschaftsraum)
- Vergleich der extrahierten Merkmale mit Merkmalen von bekannten Objekten aus einer Datenbank (Matching)

Die Methoden zur Objekterkennung lassen sich in zwei Hauptarten aufteilen: in die modellbasierte Erkennung (Kantendetektion, Houghtransformation, Snakes) und die erscheinungsbasierte Erkennung (Histogramme, Eigenspaces). Beide Arten basieren im Groben auf oben vorgestelltem Aufbau, sind aber von ihrer Vorgehensweise völlig verschieden.
Modellbasierte Objekterkennung verwendet geometrische Merkmale wie Ecken, Kanten oder Flächeninhalt, um Objekte in Bilder wiederzuerkennen. Die erscheinungsbasierte Erkennung verzichtet komplett auf die Nutzung geometrischer Merkmale und arbeitet ausschließlich auf nicht-geometrischen Merkmalen wie Farbe oder Reflektanz.

Im den nächsten Kapiteln werden nun einige dieser Methoden vorgestellt.

3.1 Kantendetektion

Das Ziel der Kantendetektion ist es, in einem Bild die Konturen von Objekten zu finden. Eine Kante wird dabei als Änderung der Grauwerte definiert. Je stärker die Änderung, desto höher die Kantenstärke. Auch die Kantenrichtung ist interessant. Es gibt unterschiedliche Möglichkeiten Kanten in einem Bild zu finden. Die Arbeitsschritte des sogenannten *Canny* Kantendetektors, welcher mit Gradienten arbeitet, sehen beispielsweise wie folgt aus:

3.1.1 Canny-Kantendetektor

Es wird eine zweidimensionale Funktion $f_{(x,y)}$ definiert. Der Gradient dieser Funktion ist $g_{(x,y)} = (f_x, f_y)$ und zeigt in die Richtung der höchsten Steigung. Der Betrag des Gradienten ist ein Maß für die Stärke der Steigung.Auf Bilder übertragen ergibt die Gradientenberechnung also sowohl

Kantenstärke als auch Kantenrichtung.
Ein Bild wird als diskrete zweidimensionale Funktion $f_{(x,y)}$ betrachtet, die den Grauwert des Punktes (x, y) liefert. Als nächstes wird eine Näherung für die partiellen Ableitungen f_x und f_y benötigt. Diese lässt sich durch eine Faltung mit dem *Sobel*-Operator bestimmen, der das Bild mit den Masken $\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$ und $\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & -0 \\ -1 & -2 & -1 \end{bmatrix}$ faltet.
Für jeden Bildpunkt (x, y) erhält man dann mit $|s(x, y)| = \sqrt{f_x^2(x, y) + f_y^2(x, y)}$ ein Maß für die Kantenstärke. Mit Hilfe der Gradientenrichtung, die sich aus $d(x, y) = arctan(\frac{f_x}{f_y})$ berechnen lässt, erhält man die Kantenrichtung des Punktes (x, y).

Folgende Schritte führen zu einer Verbesserung der Kantendetektion:

- Glättung des Ursprungsbildes vor der Kantendetektion $\rightarrow$ Reduktion von Störungen
- Verdünnung der Kanten $\rightarrow$ bessere Repräsentation von Konturlinien
- Binarisierung des Kantenbildes mittels einer Schwellwertoperation $\rightarrow$ Reduktion auf Kantenpunkte gleicher Stärke

3.2 Hough-Transformation

Die Houghtransformation baut auf der Kantendetektion auf und sucht Formen, die von Kantenpunkten gebildet werden. Dabei ist die Houghtransformation sehr allgemein verwendbar. Die in einem binarisierten Bild erkannten geometrischen Strukturen können Linien, Kreise, Ellipsen oder andere geometrische Figuren sein. Die Hough-Transformation ist sehr robust gegenüber Bildstörungen und teilweisen Verdeckungen der Objekte.

Das Grundprinzip der Houghtransformation soll nun veranschaulicht werden:
Es werden alle Bildpunkte die auf einer Kante liegen auf Hinweise untersucht, die auf eine Form deuten, die man detektieren möchte. Zu jedem Kantenpunkt werden alle möglichen Parameter in einem sogenannten Akkumulatorraum gespeichert. Dabei entspricht jeder Punkt in Akkumulatorraum einem Objekt im Bildraum. Diese Parameter könnten zum Beispiel bei Geraden die Steigung und der y-Achsen-Abschnitt sein, oder bei einem Kreis der Mittelpunkt und der Radius.
Nachdem alle Kantenpunkte untersucht wurden, werden die gesammelten Hinweise im Akkumulatorraum ausgewertet.Die Untersuchung der

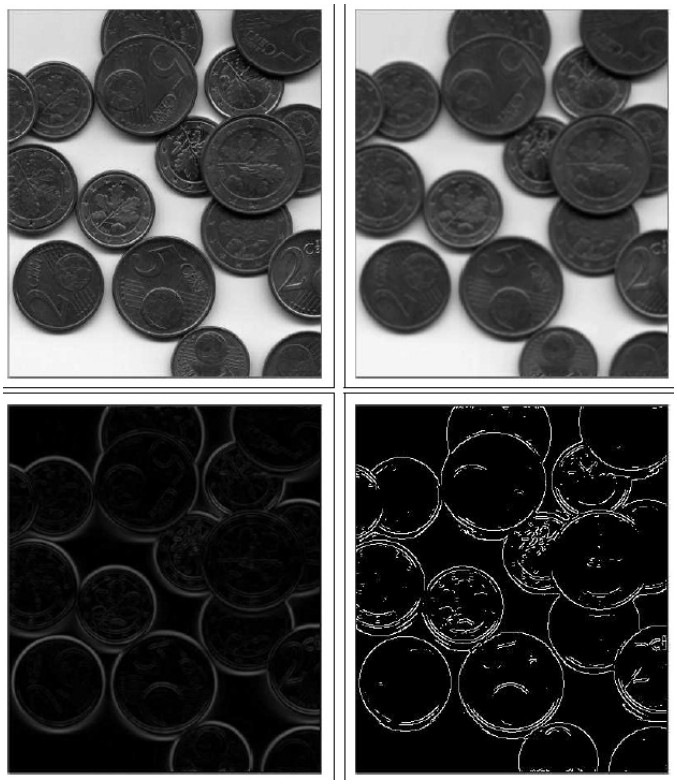

Abbildung 2: Einzelne Schritte der Canny-Kantendetektion

Kantenpunkte und die Dimension des Akkumulatorraums hängen dabei von der gesuchten Form ab.
Mit der verallgemeinerten Houghtransformation können sogar beliebige Formen, für die keine mathematische Beschreibung vorliegt, detektiert werden. Zur Beschreibung der Form genügt eine Folge von diskreten Punkten.

3.3 Histogramme

Die Objekterkennung anhand von Histogrammen stellt einen erscheinungsbasierten Ansatz dar, der ausschließlich die farblichen Eigenschaften eines Objektes zur Erkennung nutzt. Bei Objekten mit komplexen Oberflächen, auf denen sich geometrische Merkmale wie Ecken und Kanten nur schwer extrahieren lassen, eignet sich diese farbbasierte Erkennungsmethode hervorragend.

Die Verwaltung der Farben erfolgt in einem Histogramm. Ein Farbhistogramm speichert die Anzahl von Pixeln eines Bildes, die alle einen bestimmten gleichen Farbwert besitzen. Da diese Histogramme bei Be-

achtung der vollen Farbtiefe ziemlich groß werden könnten, werden die Farben meist zu Klassen gleicher Farben zusammengefasst. Diese lässt sich zum Beispiel mit dem *k-means Clustering* erreichen. Das k-means Clustering extrahiert die repräsentativsten Farben und verringert somit die Größe des Histogramms sowie die Anfälligkeit gegenüber Beleuchtungsveränderungen.
Beim Erkennungsprozess wird nun einfach das Histogramm eines Objektes mit dem des unbekannten Bildes mittels Histogramm Intersection verglichen. Als Ergebnis erhält man den Grad der Übereinstimmung zwischen den Histogrammen.
Da sich der Grad der Übereinstimmung bei abweichender Skalierung oder Orientierung des Objektes im Bild (und sogar bei teilweiser Objektverdeckung) nur geringfügig verändert, ist eine zuverlässige Objekterkennung trotzdem möglich.
Erstellt man nun eine geringe Anzahl an Histogrammen (ca. 6) von einem Objekt, welches aus verschiedenen Blickwinkeln aufgenommen wurde, ist es sogar möglich ein dreidimensionales Objekt zu erkennen.

Die Vorteil der Histogramm-basierten Erkennung sind also ganz klar die Invarianz gegenüber Skalierung, Rotation, Translation und teilweise auch gegenüber Verdeckungen. Die leichte Implementierung der Methode sowie das gute Laufzeitverhalten sind ein weiteres Plus.
Problematisch wird die Objekterkennung mit Histogrammen jedoch, sobald sich die Beleuchtung ändert. Ändert sich die Beleuchtung der Szene, so ändert sich auch die Farbe des Objektes. Und da die Farbe das Hauptvergleichsmerkmal dieser Methode ist, sinkt dementsprechend die Erkennungsrate erheblich. Ein weiteres Problem ist, dass komplett unterschiedliche Objekte (im Bezug auf ihre Geometrie) das gleiche Histogramm haben können.
Es gibt jedoch auch Ansätze, wie die des „Color Cooccurrence Histogram“, die einen Teil der Geometrie eines Objektes in das Histogramm integrieren, und somit eine Kombination aus modellbasierter und erscheinungsbasierter Objekterkennung schaffen.

3.4 Matching

Da jedes Verfahren der Objekterkennung verschiedene Merkmale extrahiert, braucht jedes Verfahren eigene Methoden zum Vergleichen von Objektbildern und Suchbildern.
Einige erscheinungsbasierte Methoden erzeugen zum Beispiel für jedes Objekt Trainingsbilder, aus denen ein mehrdimensionaler Vektor in einem Merkmalsraum gebildet wird. Mehrere Bilder desselben Objekts in verschiedenen Lagen bilden eine Punktwolke in diesem Merkmalsraum. Verschiedene Objekte bilden verschiedene Punktwolken. Da der aus

dem Suchbild erstellte Merkmalsvektor ebenfalls in diesem Raum liegt, kann die Punktwolke des Objektes anhand eines Ähnlichkeitsmaßes den Punktwolken des Suchbildes zugeordnet werden.
Bei modellbasierten Methoden kann zum Beispiel ein Attributgraph, in dem jeder Knoten einen Teil des Objekts beschreibt, ein Modell eines Objekts sein.

Es gibt aber auch Objekterkennungssysteme, die im Vergleich zu den bisher vorgestellten Systemen keine konkreten Objekte im Voraus lernen, sondern aufgrund einer Wissensbasis entscheiden, ob sich ein Objekt im Suchbild befindet.
Diese Wissensbasis kann zum Beispiel aus einem künstlichen neuronalen Netz bestehen, welches zuerst trainiert werden muss. Dazu werden dem System einige Trainingsbilder vorgelegt zu denen dann Ausgaben berechnet werden. Aufgrund dieser Ausgaben und den tatsächlichen Ergebnissen kann das System sein Neuronales Netz anpassen und verbessern. Mit Testbilder kann der „Lernerfolg“ überprüft werden. Somit besteht das Wissen des Systems nicht aus gespeicherten Merkmalen, wie bei den anderen vorgestellten Methoden, sondern im trainierten Neuronalen Netz.

Im folgenden Abschnitt wird nun das Template Matching Verfahren vorgestellt, welches in dieser Studienarbeit als Verfahren zur Mustererkennung gewählt wurde.

3.4.1 Template Matching

Das Template Matching beschreibt eine Technik in der digitalen Bildverarbeitung, die in einem Bild einen Ausschnitt sucht, der möglichst stark mit einem gegebenen Template (Objektbild) übereinstimmt. Dabei wird das Template wie eine Schablone über das gesamte Bild „gefahren“. Die Ergebniswerte zu jeder einzelnen abgetasteten Region werden abgespeichert.

Als Ergebnis erhält man ein Ähnlichkeitsmaß welches Aufschluss darüber gibt, wie wahrscheinlich es ist, dass es sich bei der gerade abgetasteten Region um den gesuchten Bildausschnitt handelt. Die Ergebnisse werden als Gleitkommazahlen in einem Bild der Größe $(img \rightarrow width - templatesize.x + 1, img \rightarrow height - templatesize.y + 1)$ gespeichert. Es gibt verschiedene Matching-Methoden die dabei angewendet werden können.

Die „Square difference matching“ Methode vergleicht die Quadrate der Differenzen zwischen Template und Bild, so dass eine absolute Übereinstimmung $Null$ als Ergebnis liefern würde und geringe Übereinstimmungen dementsprechend größer 0 wären:

Abbildung 3: Prinzip des Template Matchings

$$R_{sqdiff}(x,y) = \sum_{x',y'} \left[T(x',y') - I(x+x',y+y')\right]^2 \tag{1}$$

Bei der „Correlation matching“ Methode wird das Template mit dem Bild multipliziert. Dabei erhält man für eine gute Übereinstimmung größere Werte und für schlechtere oder keine Übereinstimmungen einen kleineren Wert oder 0.

$$R_{ccorr}(x,y) = \sum_{x',y'} \left[T(x',y') \cdot I(x+x',y+y')\right]^2 \tag{2}$$

Die „Correlation coefficient matching“ Methode vergleicht das Template mit dem Bild unter Einbeziehung der jeweiligen Mittelwerte der Bilddateien. Absolute Übereinstimmung liefert 1 als Ergebnis und absolute Diskrepanz liefert -1. Ein Wert von 0 sagt aus, dass es keine relevanten Übereinstimmungen gibt.

$$R_{ccoff}(x,y) = \sum_{x',y'} \left[T'(x',y') \cdot I'(x+x',y+y')\right]^2 \tag{3}$$

$$T'(x',y') = T(x',y') - \frac{1}{(w \cdot h) \sum_{x'',y''} T(x'',y'')} \tag{4}$$

$$I'(x+x',y+y') = I(x+x',y+y') - \frac{1}{(w \cdot h) \sum_{x'',y''} I(x+x'',y+y'')} \tag{5}$$

Jede dieser Gleichungen gibt es auch in normalisierter Form, welche die Wirkung von Beleuchtungsdifferenzen zwischen dem Template und dem Bild reduzieren können. Bei jeder Gleichung in normalisierter Form sieht der Normalisierungs-Koeffizient gleich aus:

$$Z(x,y) = \sqrt{\sum_{x',y'} T(x',y')^2 \cdot \sum_{x',y'} I(x+x',y+y')^2} \tag{6}$$

Daraus resultieren folgende Gleichungen für die drei vorgestellten Methoden:

$$R_{sqdiff_normed}(x,y) = \frac{R_{sqdiff}(x,y}{Z(x,y)} \tag{7}$$

$$R_{ccorr_normed}(x,y) = \frac{R_{ccorr}(x,y}{Z(x,y)} \tag{8}$$

$$R_{ccoeff_normed}(x,y) = \frac{R_{ccoeff}(x,y}{Z(x,y)} \tag{9}$$

4 Tracking

Das optische Tracking ist der wichtigste Bestandteil jeder AR-Anwendung. Es umfasst alle Bearbeitungsschritte die zur Verfolgung und Erkennung von statischen oder bewegten realen Objekten in Echtzeit nötig sind. Das Verfahren ermöglicht es aus Aufnahmen der realen Welt Merkmale von Objekten zu extrahieren und ausgehend davon die Position und Orientierung der Kamera zu berechnen. Mit den gewonnenen Informationen ist es möglich nun Zusatzinformationen oder virtuelle Objekte über das reale Sichtfeld einzublenden. Optische Trackingverfahren lassen sich in zwei Arten aufteilen: markerbasiertes und markerloses Tracking.

4.1 Markerbasiertes Tracking

Das markerbasierte Tracking ist ein Verfahren, das sogenannte "Markeräls Orientierungspunkte nutzt. Diese Marker können durch Verfahren der Bildverarbeitung leicht extrahiert werden und lassen durch Ihre quadratische Form eine schnelle Bestimmung der Positionierung in der realen Umgebung zu.

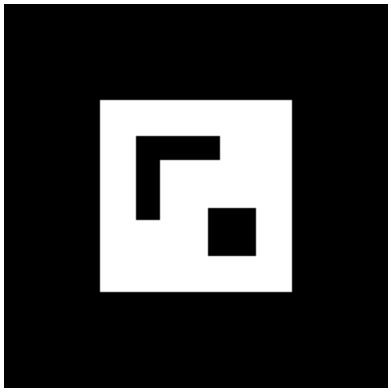

Abbildung 4: Ein AR Marker

Der Algorithmus dafür sieht dabei in der Regel wie folgt aus: Das Videobild wird nach allen Regionen durchsucht, deren Konturen durch ein Viereck beschrieben werden können. Um dies zu vereinfachen wird das Bild vorher mittels eines Schwellwertes in ein Binärbild umgewandelt. Werden die Konturen eines Markers (bzw. die des schwarzen Rahmens) detektiert, wird die Position und Ausrichtung des Markers relativ zur Kamera berechnet und eine perspektivische Transformation vollzogen, die die Eckpunkte der Marker in eine quadratische, perspektivisch korrekte Form bringen. Der Marker wird nun auf die Größe der antrainierten Marker skaliert. Anschließend kann das Muster des Markers mit denen der bekannten Marker verglichen werden.

Wird ein Marker erfolgreich erkannt, werden dessen ursprünglichen Konturlinien als Vektoren interpretiert, die eine x-y-Ebene aufspannen. Zur Erzeugung eines dreidimensionalen Raumes wird noch ein weiter Vektor berechnet, der senkrecht auf der Ebene steht. Um eine Größeneinheit für das Koordinatensystem im Raum zu erzeugen wird die Länge der Vektoren mit den Abmessungen der virtuellen Elemente, die in das Videobild integriert werden sollen, in Beziehung gebracht. Nun ist es möglich die virtuellen Objekte korrekt in der realen Umgebung einzublenden. Das markerbasierte Tracking ist aufgrund der leicht erkennbaren und eindeutigen Marker sehr schnell und robust. Allerdings ist das Verfahren sehr anfällig gegenüber Verdeckung. Auch wenn nur kleine Teile des Markers verdeckt werden kann es vorkommen, dass der Marker nicht mehr erkannt wird. Eines der bekanntesten markerbasierten Frameworks ist das *ARToolkit*.

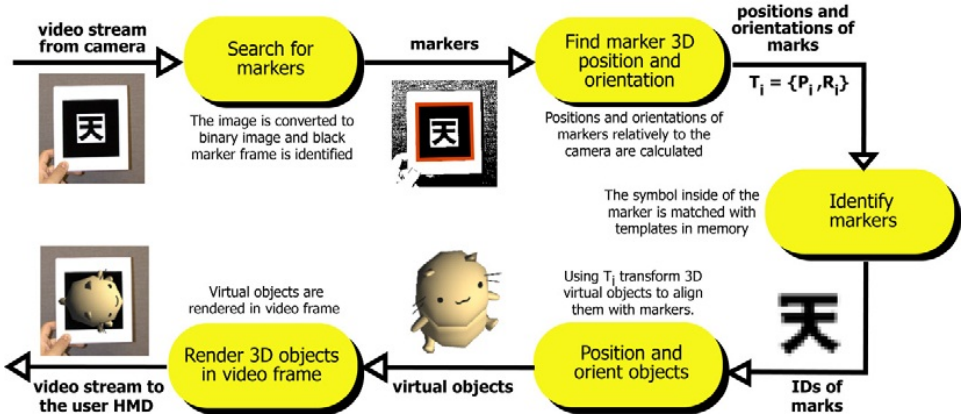

Abbildung 5: Marker-Erkennung im ARToolkit

4.2 Markerloses Tracking

Beim markerlosen Tracking werden anstatt der üblichen, künstlichen Marker natürliche Merkmale (Features oder Keypoints) der realen Umgebung benutzt um die Position der Kamera in der Welt zu bestimmen. Merkmale können markante Punkte, Kanten von Objekten aber auch Farben sein. Diese Features sind skalierungs- und rotationsinvariant, sowie teilweise invariant gegenüber Änderung der Beleuchtung oder des Blickwinkels. Die extrahierten Merkmale sind sehr markant, so dass sie mit einer hohen Wahrscheinlichkeit in weiteren Bildern wiedererkannt werden können. Um eine robuste Wiedererkennung zu gewährleisten sind mindestens 2 bis 3 Referenzbilder nötig. Jedes der Livebilder wird später mit diesen Referenzbildern verglichen. Das Verfolgen dieser Features in aufeinander folgenden Frames nennt man Feature Tracking.

4.2.1 SIFT

Das 1999 von David Lowe entwickelte featurebasierte Tracking-Verfahren SIFT (Scale-invariant feature transform) ist einer der bekanntesten Algorithmen zur Extraktion von Merkmalen aus diskreten Pixelbildern. Das Verfahren ist die Grundlage vieler Feature Tracking Algorithmen.
Der Algorithmus zur Extraktion der Merkmale eines Bildes besteht im wesentlichen aus vier Schritten:

- Scale-space extrema detection
- Keypoint localization
- Orientation assignment
- Generation of keypoint descriptors

Der erste Schritt zur Detektion interessanter Merkmale (Keypoints) ist es, verschiedene Skalierungen des Bildes mit einem Gaußfilter zu glätten und das *Difference of Gaussian*-Verfahren (DoG) mit je zwei benachbarten Bildern durchzuführen. Die DoG-Bilder werden nun nach interessanten, skalierungs- und orientierungsinvarianten, Merkmalen durchsucht. Jeder Pixel der DoG-Bilder wird sowohl mit seinen 8 Nachbarn als auch mit den jeweils 9 Nachbarn der benachbarten, unterschiedlich skalierten DoG-Bilder verglichen. Dabei gelten Pixel, die als lokales Extrema erkannt werden, als potentielle Merkmale. Aus diesen werden die Besten ausgewählt. Als Maß gilt die Stabilität des Merkmals.
Um die Ausrichtung des Keypoints zu bestimmen, werden die lokalen Bildgradienten berechnet. Dies geschieht in einer $16x16$ Nachbarschaft des Gauss-Bildes, dessen Skalierung der Skalierung des Keypoints am nächsten ist.
Aus diesen Bildgradienten werden vier Orientierungshistogramm errechnet, welche den 128-dimensionalen Vektor bilden, der zur Beschreibung des Keypoints dient. Um die Invarianz gegenüber Helligkeitsveränderungen zu erhöhen wird der Vektor noch normiert.

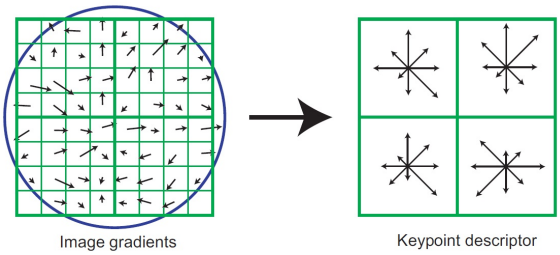

Abbildung 6: SIFT Gradientenbestimmung

Nachdem alle Keypoints von mindestens zwei bis drei Referenzbildern eindeutig bestimmt sind können sie nun mit den Livebildern der Kamera verglichen werden. Beim sogenannten Matching werden die erkannten Merkmale der Referenzbilder den entsprechenden Merkmalen der Livebilder zugeordnet. Als Maß dafür wird der Euklidische Abstand der Punkte zueinander verwendet. Zur Bewertung des Abstands wird dabei ein globaler Schwellwert für den Abstand zwischen den Merkmalspunkten, sowie das Verhältnis zwischen dem kleinsten und dem nächst kleinsten Abstand der jeweiligen Merkmalspaare zur Hilfe genommen.
Nun müssen noch Bilder als Ganzes zueinander zugeordnet werden. Dabei gilt die Summe aller korrekt zugeordneten Merkmalspunkte zwischen dem Referenzbild und dem Livebild, als auch der durchschnittliche Abstand aller zugeordneten Merkmalspunkte beider Bilder als Maß für eine Bewertung. Hier wird für jeden Referenzbild zum Einen die komplette Anzahl aller zugeordneten Punkte zwischen ihm und dem aktuellen Bild berechnet. Zum Anderen wird der durchschnittliche Abstand zwischen allen zugeordneten Punkten dieser beiden Frames gebildet.Das Referenzbild, das die höchste Anzahl an zugeordneten Punkten zum aktuellen Bild und den kleinsten durchschnittlichen Abstand der Merkmale zueinander aufweisen wird als am meisten geeignet bestimmt.

5 Implementierung und Methoden

Die Studienarbeit wurde in der Programmiersprache C++ unter Verwendung der Programmbibliothek OpenCV geschrieben. OpenCV bietet viele Funktionen und auch Klassen aus dem Bereich der Echtzeit-Computergrafik und Bildverarbeitung an. Vor allem eine große Auswahl an Funktionen zur Objekterkennung und Bildübertragung waren für die Implementierung vom Vorteil.

5.1 Spielkartenerkennung

5.1.1 Rechtecke finden

Um die einzelnen Spielkarten zu extrahieren müssen zunächst die Konturen im Bild gefunden werden. Konturen sind Listen von Punkten die eine Kurve im Bild darstellen. Diese Punkte werden in sogenannten Sequenzen abgespeichert, bei denen jeder Eintrag Informationen über den nächsten Punkt auf der Kurve enthält. Konturen lassen sich in OpenCV mittels cvFindContours() finden. Als Eingabe sind Binärbilder erlaubt, die zum Beispiel mit cvCanny() (liefert als Ergebnis Kantenbilder der Canny-Kantendetektion), cvThreshold() oder cvAdaptiveThreshold() (liefern als Ergebnis Kantenbilder, deren Kanten Grenzen zwischen negativen und positiven Regionen darstellen) erstellt wurden. Dabei wurde cvThreshold()

als Methode zur Binarisierung gewählt. Somit muss nun zunächst, um Konturen im Bild zu finden, das Ausgangsbild aus dem RGB-Farbraum in ein Graubild umgewandelt werden. Zur Reduktion des Rauschens wird das Bild verkleinert und anschließend wieder vergrößert. Nun wird das Bild noch mittels cvThreshold() binarisiert und das Ergebnis an cVFind-Contours() weitergegeben. Als Resultat erhält man eine Sequenz aus Konturen, aus denen jetzt diejenigen rausgefiltert werden müssen, die die Eigenschaften einer Spielkarte aufweisen. Das heißt: vier Kanten bzw. vier Eckpunkten, konvexe Form, und eine "relativ"große Fläche.
Da die Konturen noch aus einzelnen, benachbarten Pixeln bestehen, müssen sie zuerst auf ihre markantesten Punkte reduziert werden, was bei einer Spielkarte den vier Eckpunkten entspricht. Dies wird erreicht durch cvApproxPoly(). Diese Funktion wendet die *Douglas-Peucker* Approximation auf die aktuelle Kontur an. Die Spielkarten haben allerdings abgerundete Ecken. Das führt dazu, dass die vier Punkte, die die Karte beschreiben genau dort liegen, wo die abgerundeten Ecken beginnen, da hier der Unterschied der Vektoren der benachbarten Pixel am größten ist. Das führt dazu, dass Teile der Spielkarte nicht innerhalb der Kontur liegen und zudem die Karte innerhalb der Kontur zusätzlich schief ist. Um dies zu umgehen, ist es notwendig eine Bounding Box um die approximierte Kontur zu legen, die die komplette Karte umfasst. Um zu verhindern, dass die Bounding Box aufgrund einer zu starken perspektivischen Verzerrung (durch eine nicht senkrecht nach unten gerichtete Kamera) einen Bereich eingrenzt, der nicht zur Karte gehört, müssen zunächst noch die Winkel zwischen den anliegenden Kanten überprüft werden. Entsprechen alle Winkel ungefähr 90 Grad, wird mit cvMinAreaRect2() eine Bounding Box zurückgegeben, die die Kontur einschließt. Dabei liefert cvMinAreaRect2(), im Gegensatz zu cvBoundingRect() (gibt ein Rechteck zurück, das die Kontur einschließt und parallel zu x- und y-Achse ist), das kleinst mögliche Rechteck, welches die Kontur einschließt. Sollten ein oder mehr Winkel zu stark von 90 Grad abweichen, werden die ursprünglichen Eckpunkte der Approximation verwendet. Die Eckpunkte der der erkannten Spielkarten werden in einer neuen Sequenz gespeichert.

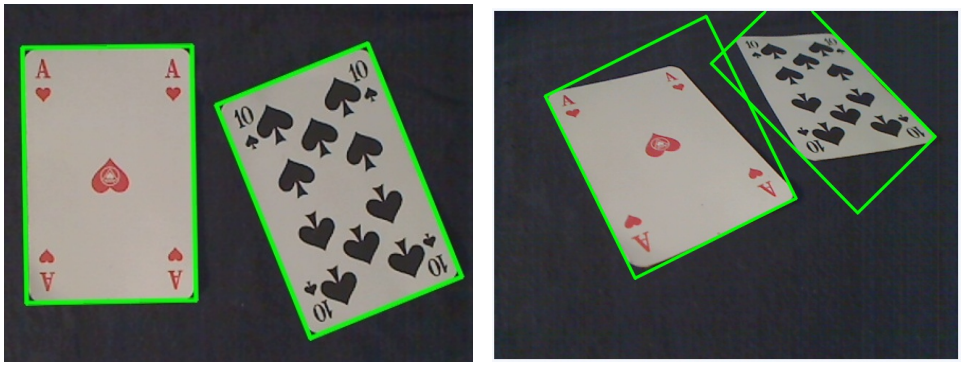

Abbildung 7: Erkannte Karten mit Boundingbox

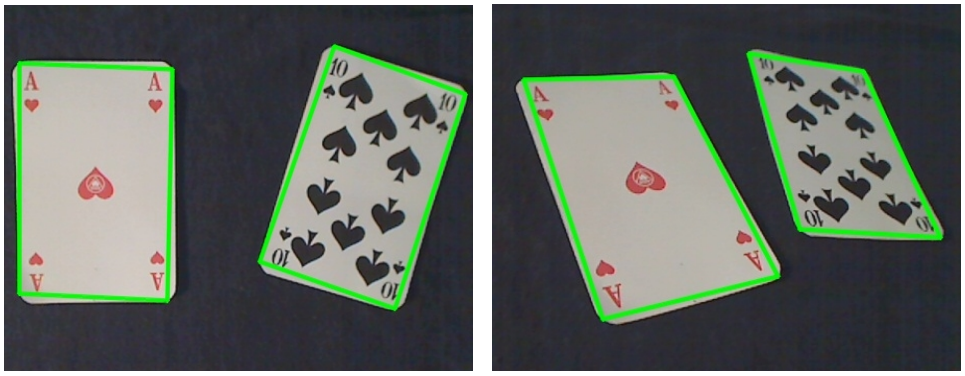

Abbildung 8: Erkannte Karten ohne Boundingbox

5.1.2 Spielkarten-Extraktion

Nun gilt es mithilfe dieser Sequenz die einzelnen Spielkarten zu extrahieren. Um später den Wert und die Farbe der Karte mithilfe des Template Matching bestimmen zu können, ist es notwendig die Karten einheitlich zu rotieren, zu skalieren und perspektivisch korrekt darzustellen.
Um sicherzustellen, dass alle Karten am Ende eine einheitlich hochkante Ausrichtung haben, müssen zunächst die Eckpunkte der Boundingbox auf Ihre Reihenfolge untersucht werden. Dazu wird der Betrag des Vektors vom ersten und zweiten Eckpunkt mit dem Betrag des Vektors vom zweiten und dritten Eckpunkt verglichen. Ist der erste Vektor kleiner als der zweite müssen die Eckpunkte innerhalb der Kontur rotiert werden.
Nun ist sichergestellt, dass alle Karten im richtigen Format transformiert werden können. Es wird eine Bilddatei mit den Abmessungen von 300 x 450 Pixeln (Seitenverhältnis einer gewöhnlichen Spielkarte) erstellt, in der die Spielkarte gespeichert werden soll. Damit die Karte korrekt in der Bilddatei abgespeichert werden kann, wird sie einer perspektivischen Transfor-

mation unterzogen.
Da die Eckpunkte der Bounding Box ein Parallelogram beschreiben würde eine affine Transformation ausreichen. Jedoch wird für die Transformation der approximierten Eckpunkte, die bei ungünstigen Kamerawinkel gewählt werden, eine homographische Transformation angewandt, da die Eckpunkte kein Parallelogramm bilden. Der wesentliche Unterschied zur affinen Transformation ist die Angabe der homografischen Transformation als 3×3 Matrix anstatt als 2×3 Matrix, weswegen letztere für beide Fälle genutzt werden.

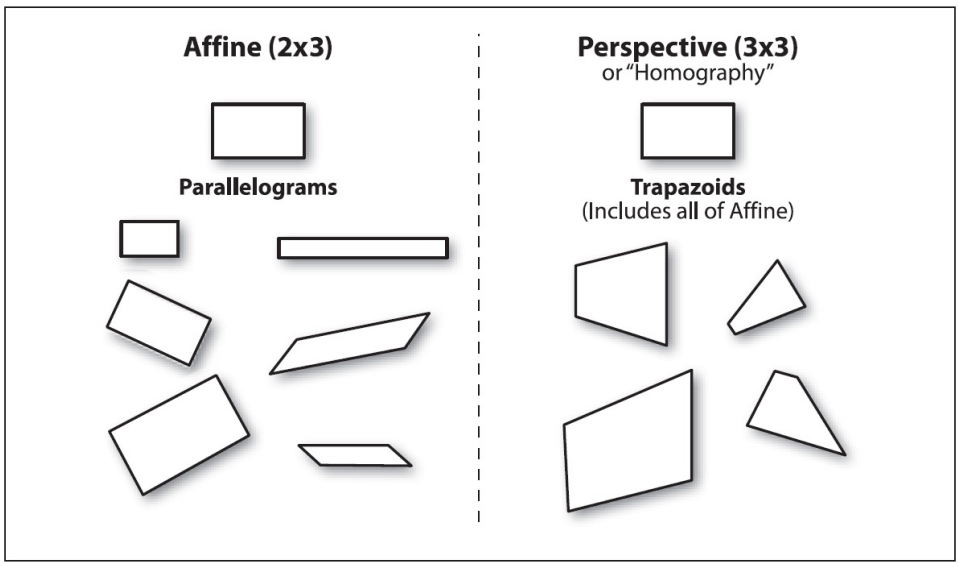

Abbildung 9: Affine und hompgraphische Transformation

Affin heisste jede Transformation, die als Matrixmultiplikation gefolgt von einer Vektoraddition beschrieben werden kann. Wie bereits erwähnt wird die Transformation durch eine $2x3$ Matrix dargestellt und wie folgt definiert:

$$A \equiv \begin{bmatrix} a_0 0 & a_0 1 \\ a_1 0 & a_1 1 \end{bmatrix} B \equiv \begin{bmatrix} b_0 \\ b_1 \end{bmatrix} T \equiv \begin{bmatrix} A & B \end{bmatrix} X \equiv \begin{bmatrix} x \\ y \end{bmatrix} X' \equiv \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{10}$$

Jedes Parallelogram $ABCD$ kann dabei durch die affine Transformation **A** · **X** + **B** in jedes beliebige andere Parallelogram $A'B'C'D'$ umgewandelt werden. Dabei lässt sich die affine Transformation auch beschreiben, indem der Vektor **X** auf den Vektor **X′** erweitert und dann eine einfache Linksmultiplikation von **X′** mit **T** durchgeführt wird.

Bei der homographischen oder perspektivischen Transformation wird

eine Ebene auf eine andere „gemappt“. Der Abbildungsprozess eines Objektes $P_{src} = [X, Y, Z, 1]^T$ auf die Bildebene $P_{dst} = [x, y, 1]^T$ kann folgendermaßen beschrieben werden:

$$P_{dst} = sH \cdot P_{src} \tag{11}$$

Parameter s steht für einen Skalierungsfaktor. Die Hompgraphie H besteht aus zwei Teilen. Zum einen aus den extrinsischen Kameraparametern W, welche sich aus der Rotation R und der Translation t zusammensetzen und zum anderen aus der Matrix M, welche die intrinsischen Parameter der Kamera beinhaltet.

$$P_{dst} = sMWP_{src} \text{ wo } M = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \text{ und } W = \begin{bmatrix} R & t \end{bmatrix} \tag{12}$$

Definiert man nun, ohne dass es der Allgemeingültigkeit widerspricht, die Ebene des Objektes so, dass $Z = 0$ ist, so kann man die Rotationsmatrix R aufsplitten auf r_1, r_2 und r_3, so dass die Ausgangsgleichung nun so aussieht:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = sM \begin{bmatrix} r_1 & r_2 & r_3 & t \end{bmatrix} \begin{bmatrix} X \\ Y \\ 0 \\ 1 \end{bmatrix} = sM \begin{bmatrix} r_1 & r_2 & t \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \tag{13}$$

Die Homographie-Matrix H, die einen Punkt aus der Objektebene in die Bildebene projiziert kann also beschrieben werden als $H = sM\,[r_1 r_2 t]$. Dabei ergibt sich für die Beziehung zwischen P_{dst} und P_{src}:

$$P_{dst} = sHP_{src} \tag{14}$$

Diese Gleichung wird nun genutzt um für jedes Paar korrespondierender Punkte jeweils eine Gleichung aufzustellen. Mindestens vier Punktpaare werden dabei vorausgesetzt um eine Homographie-Matrix berechnen zu können.
Zur Bestimmung der Homographie-Matrix sind folglich insgesamt acht Punkte aus dem Originalbild und dem Bild, in das die Karte extrahiert werden soll, zu wählen. Die Punkte werden in 2×4 Matrizen gespeichert, aus denen dann mit cvFindHomography() die Homographie-Matrix berechnet wird. Nun kann mit cvWarpPerspective() die Spielkarte aus dem Kamerabild in eine eigene Datei gemappt werden.

```
// Bild mit den Seitenverhältnissen einer Spielkarte
//wird erstellt
dst = cvCreateImage( cvSize( DST_WIDTH, DST_HEIGHT ),
  IPL_DEPTH_8U, 3 );

// BoundingBox-Eckpunkte der Karte
double a[] = {
p0x, p0y,
p1x, p1y,
p2x, p2y,
p3x, p3y
};

// Eckpunkte des erzeugten Bildes für die transfor-
//mierte Karte
double b[] = {
0, 0,
0, DST_HEIGHT - 1,
DST_WIDTH - 1, DST_HEIGHT - 1,
DST_WIDTH - 1, 0
};

//Matrizen werden erzeugt
CvMat src_point = cvMat( NUM_POINTS, 2, CV_64FC1, a );
CvMat dst_point = cvMat( NUM_POINTS, 2, CV_64FC1, b );
CvMat *h = cvCreateMat( 3, 3, CV_64FC1 );

// Eine Homographie-Matrix wird berechnet und in h
//gespeichert
cvFindHomography( &src_point, &dst_point, h );

// Mit Hilfe der Homographie-Matrix wird die Karte vom
// Ausgangsbild ins Zielbild gemappt
cvWarpPerspective( src, dst, h );
```

5.1.3 Karten-Klassifikation

Da die Karte nun extrahiert ist, gilt es die Spielkarte bezüglich ihrer Farbe und Ihres Wertes zu klassifizieren. Es gibt dazu mehrere Möglichkeiten. Aufgrund vorliegender einheitlicher Skalierung und Rotation der Karte, bietet sich das Template Matching an. Dies ist schnell und liefert gute Ergebnisse.

Template Matching

Um das Template Matching nun anzuwenden, ist es als erstes natürlich

notwendig die Templates an sich zu erstellen. Um die Pokerkarten zu erkennen, müssen von allen vier Farben und 13 Werten eines Spielkartendecks Templates erstellt werden. Je mehr Templates von den jeweiligen Farben und Werten gemacht werden, desto besser die spätere Erkennung. Es empfiehlt sich dabei dies unter verschiedenen Konditionen (Licht, Blickwinkel, etc.) vorzunehmen. Zur Erstellung der Templates wird eine Speicherfunktion genutzt, die auf Knopfdruck die extrahierten Karten in eine externen Bilddatei auf dem Rechner speichern kann. Somit ist es auch zu einem späteren Zeitpunkt möglich, bei einer Fehlerkennung aus der Spielkarte ein Template zu generieren, so dass die Template-Datenbank bei jedem Fehler erweitert werden kann. Aus den gespeicherten Bilddateien kann nun, mittels eines Bildbearbeitungsprogramms, die Spielkarte nur auf Ihre Farbe oder ihren Wert reduziert werden.

Sobald eine kleine Datenbank aus Templates angelegt wurde, können nun die Livebilder der Kamera auf die Templates untersucht werden. Dabei werden die Karten separat auf Farbe und Wert untersucht. Die Funktionen sind bis auf die Benutzung unterschiedlicher Templates fast identisch.
Um den unnötigen Rechenaufwand zu verringern, der beim Vergleichen des gesamten Bildes mit über 50 Templates entstehen würde, wird der zu untersuchende Bildbereich stark eingeschränkt. Da die linke obere Ecke jeder Spielkarte identisch mit der rechten unteren Ecke ist (nach einer Rotation um 180°), und sowohl die Farbe und der Wert der Karte enthalten sind, ist dieser Bereich bestens geeignet. Außerdem unterscheiden sich die Symbole für die Farbe auf Karten mit Bildern und den restlichen Karten in ihrer Größe. Da das Template Matching Verfahren nicht invariant gegen Skalierungsunterschiede ist, würde das zu fehlerhaften Ergebnissen führen.

Nun wird also mit cvSetImageROI() die „Region of Interest“ auf den linken oberen Kartenbereich gelegt. Das zu untersuchende Bild und das aktuelle Template werden nun in Grauwertbilder umgewandelt und danach mit Hilfe des Thresholding Verfahrens binarisiert. Nachdem nun ein Bildvektor der Größe $(img \rightarrow width - templatesize.x + 1, img \rightarrow height - templatesize.y + 1)$ zum Speichern der Template-Matching-Ergbnisse angelegt wurde, können nun das Template und das zu untersuchende Bild miteinander verglichen werden. Der Aufruf der Funktion erfolgt über cvMatchTemplate(). Dabei hat sich nach mehreren Testläufen die oben vorgestellte normierte „Square difference matching“-Methode am geeignetesten herausgestellt. Die anderen Methoden waren vergleichsweise langsam oder lieferten auch nicht die gewünschten Resultate.
Als nächstes wird aus dem Bildvektor, der jetzt die Ergebnisse des Template Matching enthält, der Eintrag mit dem kleinsten Ergebnis gesucht.

Dieser repräsentiert die Stelle, an der das Template die höchste Übereinstimmung mit dem Bild hatte. Der entsprechende Wert, der ein Maß für die Größe der Übereinstimmung ist, wird extrahiert und als aktuell bestes Ergebnis gesetzt.
Dieser Algorithmus wird nun mit allen Templates der untersuchten Karteneigenschaft (Farbe oder Wert) wiederholt. Dabei wird jedes mal überprüft, ob das Ergebnis des aktuellen Templates besser als die des vorherigen ist.Somit lässt sich bestimmen, welches Template die besten Ergebnisse liefert und damit auch welche Farbe und welchen Wert die Karte hat.

Abbildung 10: Erkannte Templates

5.2 Wahrscheinlichkeitsberechnung

5.2.1 Erste Ansätze zur Wahrscheinlichkeitsberechnung

Mit diesen Informationen ist es nun möglich Wahrscheinlichkeiten für die erkannten Karten zu berechnunen. Dabei stellt sich die Frage nach der allgemeinen Vorgehensweise. Es würde natürlich nahe liegen die Wahrscheinlichkeiten mittels mathematischer Wahrscheinlichkeitsrechnungen zu ermitteln. Bei verschiedenen Glücksspielen, wie beispielsweise beim Lotto, ist dies auch sinnvoll. Allerdings im Bezug auf Texas Hold'Em wur-

de dieser Ansatz nach langen Überlegungen jedoch verworfen. Das hatte mehrere Gründe. Vor allem das Regelwerk, welches für die Wahrscheinlichkeitsberechnung erstellt werden müsste wäre sehr umfangreich und komplex. Für jede Handart (Paar, Drilling, Straße) müsste ein eigenes, kleines Regelwerk erstellt werden. Diese Regelwerke unterscheiden sich zudem nochmal je nachdem ob 3 oder 4 Gemeinschaftskarten aufgedeckt wurden. Also verdoppelt sich die Menge der Regeln. Außerdem wären sehr viele Ausnahmeregelungen zu beachten. Anhand des Beispiels in Abbildung **??** soll dies verdeutlicht werden.

Verfügbare Karten:	**Kreuz 9 – Kreuz 10 – Pik 10 – Pik Bube – Pik Dame – Pik Ass**	
Entstehende Hand	**Benötigte Karten**	**Ausnahmen**
Höchste Karte	/	Bedingung für Hand kann nicht erfüllt werden
Paar	egal	Keine 10 → Drilling Kein Pik → Flush Keine Ass oder 8 → Straße
Zwei Paare	/	Bedingung für Hand kann nicht erfüllt werden
Drilling	10	/
Straße	9 Ass	Keine Pik 9 → Straigt Flush Kein Pik Ass → Royal Flush
Flush	Pik	Keine Pik 9 → Straigt Flush Kein Pik Ass → Royal Flush
Full House	/	Bedingung für Hand kann nicht erfüllt werden
Vierling	/	Bedingung für Hand kann nicht erfüllt werden
Straight Flush	Ass	kein Pik Ass → Royal Flush
Royal Flush	Pik Ass	/

Abbildung 11: Beispiel für Ausnahmeregelungen

Dies läßt die Komplexität, die durch einige Kartenkonstellationen entstehen können, erahnen. Außerdem ist erkennbar, dass die Regeln miteinander verknüpft werden müssten.
Nach Erstellung der ersten, einfachen Regeln musste jedoch festgestellt werden, dass es sogar Regeln gibt, die mit den normalen Rechenarten der Wahrscheinlichkeitsrechnung nicht ohne Weiteres berechnet werden können (z.B. die Berechnung der Wahrscheinlichkeit einer Verbesserung auf gleicher Ebene: ein Straße wird zu einer besseren Straße). Somit schien dieser Ansatz, vor allem unter Berücksichtigung des Zeitplanes für diese Studienarbeit, als komplett ungeeignet.
Also wurde ein Ansatz benötigt, der auf andere Weise erlaubt Wahrscheinlichkeiten zu berechnen. Der nächste Ansatz, der im Endeffekt zum gewünschten Ergebnis führte, weicht von der Idee ab, die Wahrscheinlichkeiten nur aufgrund der gegebenen Karten zu berechnen.
Eine Liste aus allen möglichen Kombinationen von 5 Karten wird pro Ein-

trag nach Vorkommen der vorhandenen Karten durchsucht. Dadurch lassen sich alle Hände finden, die sich aus den vorhandenen Karten und den noch nicht aufgedeckten Gemeinschaftskarten bilden lassen. Diese Hände werden dann auf ihre Werte untersucht und im Anschluss werden die Wahrscheinlichkeiten berechnet. Allerdings wurde im Verlauf der Implementierung auf Basis einer ähnlichen Grundidee eine Möglichkeit gefunden, ohne eine Liste aller existierenden Handkombinationen zum gewünschten Ergebnis zu kommen.

5.2.2 Bestimmung des Handwertes

Um mit den bisher gewonnen Informationen weiter arbeiten zu können, muss zuerst einmal eine grundlegende Datenstruktur definiert werde. Karten werden als *struct* aus zwei Integer-Werten definiert. Diese stehen jeweils für Farbe und Wert der Karte. Eine Hand besteht aus fünf Karten und einem Integer-Wert, der den Wert der Hand beschreibt (d.h. ob die Hand eine Straße ist, oder ein Drilling etc.).
Um diesen Handwert zu ermitteln wird ein Regelwerk aus Funktionen benötigt. Das Vorgehen, mit dem die wird nun grob skizziert:

- **Paar:** Prinzip: Es wird die Häufigkeit der Vorkommen der einzelnen Werte der Karten bestimmt. Bedingung: Ein Wert kommt zwei mal vor.

- **Zwei Paar:** Prinzip: Es wird die Häufigkeit der Vorkommen der einzelnen Werte der Karten bestimmt. Bedingung: Zwei Werte kommen jeweils zwei mal vor.

- **Drillinge:** Prinzip: Es wird die Häufigkeit der Vorkommen der einzelnen Werte der Karten bestimmt. Bedingung: Ein Wert kommt drei mal vor.

- **Straße:** Prinzip: Die Hand wird gemäß Ihrer Werte sortiert. Es wird überprüft, ob alle Karten aufeinanderfolgende Werte haben. Bedingung: Jeder Wert der sortierten Liste ist um 1 kleiner als der folgende Wert.

- **Flush:** Prinzip: Es wird die Häufigkeit der Vorkommen der einzelnen Farben der Karten bestimmt. Bedingung: Alle Farben haben die gleiche Farbe.

- **Full House:** Prinzip: Es wird die Häufigkeit der Vorkommen der einzelnen Werte der Karten bestimmt. Bedingung: Ein Wert kommt zwei mal vor, ein anderer kommt 3 mal vor.

- **Vierling:** Prinzip: Es wird die Häufigkeit der Vorkommen der einzelnen Werte der Karten bestimmt. Bedingung: Ein Wert kommt vier mal vor.

- **Straight Flush:** Bedingung: Hand ist sowohl Straße als auch Flush.

- **Royal Straight:** Bedingung: Hand ist Straight Flush. Höchste Karte ist ein Ass.

Beim Aufruf von gethandvalue() wird eine Hand nun auf Ihren Wert überprüft. Dabei wird mithilfe der oben beschriebenen Funktionen zuerst auf den höchst möglichen Handwert (also Royal Straight), dann auf den zweit höchsten, usw. untersucht. Dadurch wird verhindert, dass zum Beispiel ein Straight Flush als Flush gewertet wird.
Da nun jede Hand bezüglich ihres Wertes klassifiziert werden kann, könnten jetzt schon verschiedene Hände miteinander verglichen werden. Dies funktioniert allerdings nur, wenn die Hände unterschiedliche Handwerte haben. Es kann zum Beispiel bei einem Vergleich von zwei Straßen oder zwei Drillingen noch nicht entschieden werden, welche Hand besser ist. Von daher wird noch eine Funktion benötigt, die eine Hand genauer analysieren kann.
Die Funktion gethandstats() gibt Informationen über die einzelnen Teile einer Hand zurück. Entsprechend des Handwertes wird zum Beispiel beim Handwert „zwei Paare“ bestimmt, welches das höchste Paar ist, welches das zweit höchste und welchen Wert die einzelne Karte hat. Aufgrund dieser Informationen kann mit der Funktion comparehands() bei Eingabe von zwei Händen genau bestimmt werden, welche dieser Hände besser ist.

5.2.3 Wahrscheinlichkeiten

Um nun die Wahrscheinlichkeiten für mögliche Hände aus einem Satz von Handkarten und Gemeinschaftskarten bestimmen zu können ist es notwendig, dass der User seine Handkarten und mindestens 3 Gemeinschaftskarten eingelesen hat. Diese werden in globalen Variablen gespeichert. Sind drei oder vier der fünf möglichen Gemeinschaftskarten aufgedeckt, wird zunächst einmal ermittelt welche die aktuelle beste Hand ist, die durch eine Kombination aus Handkarten und Gemeinschaftskarten erhalten werden kann.
Für den Fall, dass erst drei Gemeinschaftskarten aufgedeckt wurden, kann dieser Schritt vereinfacht werden. Da insgesamt nur fünf Karten bekannt sind, können diese als aktuell beste Hand gewertet werden. Nachdem nun bestimmt wurde welche Kombination aller bekannten Karten die beste Hand bildet, können nun die restlichen, noch nicht bekannten Gemeinschaftskarten für die weiteren Berechnungen herangezogen werden.

Bei vier aufgedeckten Gemeinschaftskarten wird jede Karte aus dem verbliebenen Kartendeck mit den sechs bekannten Karten verknüpft. Entsprechend werden bei drei aufgedeckten Gemeinschaftskarten alle Kombinationen von zwei Karten aus dem verbliebenen Kartendeck mit den fünf bekannten Karten verknüpft.
Aus diesen sieben Karten werden nun alle Kombinationen berechnet, die eine Hand ergeben. Die Handwerte dieser Hände werden nun mit dem Handwert der zu Beginn bestimmten aktuellen besten Hand verglichen. Diese Handwerte werden nun in einem Array akkumuliert, der die Häufigkeit der Vorkommen der einzelnen Handwerte, die höher als die der aktuellen Hand sind, enthält. Aus diesem Array können nun, nachdem alle noch möglichen Hände mit der aktuell besten Hand verglichen wurden, die exakten Wahrscheinlichkeiten ermittelt werden. Diese geben prozentuale an wie wahrscheinlich es ist mit den restlichen Gemeinschaftskarten eine bestimmte Hand zu bekommen.
Wurden vom User zwei Handkarten und fünf Gemeinschaftskarten aufgedeckt ist es natürlich unnötig Wahrscheinlichkeiten im Bezug auf seine eigene Hand zu ermitteln. Nachdem die beste Hand aus Hand- und Gemeinschaftskarten bestimmt wurde, die die finale Hand des Spielers darstellt, wird noch die Gewinnwahrscheinlichkeit dieser Hand gegen die potentiellen Hände der anderen Spieler ermittelt. Dazu werden nach dem oben beschriebenen Schema die fünf Gemeinschaftskarten mit allen im Kartendeck noch übrigen Karten kombiniert, um herauszufinden wievviele Hände es gibt, die die Hand des Spielers schlagen könnten.

5.3 Ausgabe

Dem User werden die Ergebnisse der Kartenerkennung und Wahrscheinlichkeitsberechnung in einem OpenGL-Fenster angezeigt. Dort werden die Handkarten, Gemeinschaftskarten und das aktuell beste Blatt durch entsprechende Bilddateien repräsentiert. Die berechneten Wahrscheinlichkeiten werden als Text ausgegeben. Außerdem dienen zwei Buttons zum einlesen von Hand- und Gemeinschaftskarten. Das Kamerabild wurde aufgrund einer sinnvollen Minimalgröße nicht in das OpenGL-Fenster integriert und wird in einem separaten Fenster angezeigt.

6 Ergebnisse

6.1 Hardware und Vorbereitungen

Zur Erkennung der Spielkarten diente eine hochauflösende Logitech Webcam. Durch die höhere Auflösung sind die erzielten Ergebnisse wesentlich besser als von Webcams mit Standardauflösungen. Erste Versuche mit einem HD-Camcorder wiesen verschiedene Probleme auf. Zum einen war

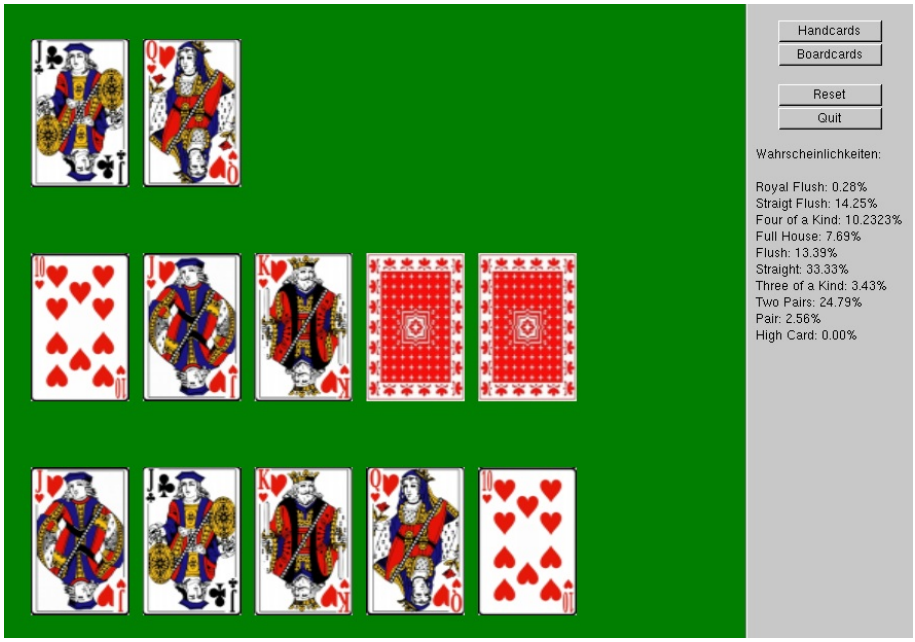

Abbildung 12: Ausgabe der erkannten Karten und Wahrscheinlichkeiten

OpenCV nicht kompatibel mit dem HDV Videoformat des Camcorders, somit war nur eine Auflösung von 640 x 480 Pixeln möglich. Des weiteren enthielten die Kamerabilder, aufgrund der Aufnahme von 50 Halbbildern pro Sekunde, deutliche Interlace-Streifen. Dies führte insbesondere bei Aufnahmen aus der Hand mit leichten Bewegungen zu unbrauchbarem Bildmaterial.
Zur Stabilisierung des Kamerabildes wurde die Webcam im Nachhinein auf einem Stativ befestigt und so justiert, dass sie aus möglichst senkrechter Position von oben mindestens fünf Spielkarten erfassen konnte. Zur besseren Differenzierung der Spielkarten vom Hintergrund eignet sich ein dunkles Tuch oder Ähnliches. Die meisten Kartendecks haben optisch unterschiedliche Werte und Farben. Von daher kann das Programm nur mit Karten durchgeführt werden, aus denen auch die Templates erstellt wurden.

6.2 Ablauf

Nachdem die Applikation aufgerufen wurde, hat der Spieler die Möglichkeit seine Karten über zwei Buttons in einem OpenGL Fenster einlesen zu lassen. Es wird jedoch eine gewisse Anzahl an Hand- und Gemeinschaftskarten vorausgesetzt, um dem Spieler Informationen über die Wahrscheinlichkeiten seiner Hand zu geben. Die von der Wahrscheinlichkeitsberechnung beachteten Kartenkombinationen sind: zwei Handkarten und

drei Gemeinschaftskarten, zwei Handkarten und vier Gemeinschaftskarten, zwei Handkarten und fünf Gemeinschaftskarten. Um zu unterscheiden was Hand- und was Gemeinschaftskarten sind, werden diese jeweils über einen eigenen Button eingelesen. Zum Einlesen der Handkarten werden genau zwei Karten benötigt. Bei den Gemeinschaftskarten können drei bis fünf Karten gleichzeitig eingelesen werden. Hat der Spieler seine Karten eingelesen, werden ihm verschiedene Informationen angezeigt. Der Spieler sieht anhand von Bilddateien, die für einzelne Spielkarten stehen, welche Karten er eingelesen hat und aus welchen Karten die beste aktuell Hand besteht.
Für den Fall von drei oder vier eingelesenen Gemeinschaftskarten werden dem User in einem Textausgabefenster prozentuale Wahrscheinlichkeiten angezeigt, die aussagen inwiefern sich die Hand des Spielers mit den verbleibenden Karten verbessern könnte. Bei fünf eingelesenen Gemeinschaftskarten werden diese Informationen nicht mehr benötigt, da sich die Hand des Spielers nicht mehr verändern kann. Anstatt dessen erhält der Spieler eine prozentuale Angabe, wie wahrscheinlich es ist, dass seine Hand besser ist als eine potentielle Hand des Gegenspielers. Diese Wahrscheinlichkeit wird anhand der fünf Gemeinschaftskarten (da der Gegner mindestens drei davon in seine Hand mit einbringen muss) und dem Wissen über die eigene Hand berechnet. Der User kann zu jedem Zeitpunkt im Programm neue Hand- oder Gemeinschaftskarten einlesen.

6.3 Genauigkeit

Da das Programm invariant gegenüber Skalierung, Rotation und leichten Veränderungen liefert es bei einem günstigen Blickwinkel gute Ergebnisse. Bei Veränderungen des Blickwinkels, so dass die Karten im Ursprungsbild perspektivisch verzerrt sind, erhöht sich die Fehlerquote jedoch.
Zur Überprüfung der Genauigkeit des Systems wird eine Testreihe unter verschiedenen Bedingungen durchgeführt. Dabei werden die Erkennungsraten bei Tageslicht, Kunstlicht, sowie Veränderungen des Blickwinkels ermittelt. Es werden jeweils vier Karten, aus einem 32 Karten umfassenden Kartendeck, gleichzeitig eingelesen. Jedes Bild wird mit 16 Templates für die Farben und 34 Templates für die Werte verglichen.

Wert/Farbe	Pik (P)		Kreuz (K)		Karo (Ka)		Herz (H)	
	Wert	Farbe	Wert	Farbe	Wert	Farbe	Wert	Farbe
Ass (A)	A	P	7	K	A	Ka	A	H
König (K)	K	P	K	K	K	Ka	K	H
Dame (D)	D	P	D	K	D	Ka	D	H
Bube (B)	B	P	B	K	B	Ka	B	H
10	10	P	10	K	10	Ka	10	H
9	9	P	9	K	9	Ka	9	H
8	8	P	8	K	8	Ka	8	H
7	7	P	7	K	7	Ka	7	H

	Total:
Erkannte Werte	96,88%
Erkannte Farben	100,00%
Insgesamt erkannt	98,44%
Insgesamt korrekt erkannte Karten	96,88%

Abbildung 13: Erkennungsraten bei Tageslicht

Wert/Farbe	Pik (P)		Kreuz (K)		Karo (Ka)		Herz (H)	
	Wert	Farbe	Wert	Farbe	Wert	Farbe	Wert	Farbe
Ass (A)	A	P	7	K	A	Ka	A	H
König (K)	K	P	K	K	K	Ka	K	H
Dame (D)	D	P	D	K	D	Ka	D	H
Bube (B)	7	P	B	K	B	Ka	B	H
10	10	P	10	K	10	Ka	10	H
9	9	P	9	K	9	Ka	9	H
8	8	P	8	K	8	Ka	8	H
7	7	P	7	K	7	Ka	7	H

	Total:
Erkannte Werte	96,88%
Erkannte Farben	100,00%
Insgesamt erkannt	98,44%
Insgesamt korrekt erkannte Karten	96,88%

Abbildung 14: Erkennungsraten bei Kunstlicht

Wert/Farbe	Pik (P)		Kreuz (K)		Karo (Ka)		Herz (H)	
	Wert	Farbe	Wert	Farbe	Wert	Farbe	Wert	Farbe
Ass (A)	7	P	A	K	A	Ka	A	H
König (K)	7	P	7	Ka	7	Ka	K	Ka
Dame (D)	7	Ka	7	Ka	7	Ka	7	Ka
Bube (B)	7	P	7	Ka	7	Ka	7	Ka
10	7	P	7	Ka	7	Ka	10	H
9	7	Ka	7	K	7	Ka	7	Ka
8	8	Ka	8	K	7	Ka	7	Ka
7	7	K	7	K	7	Ka	7	H

	Total:
Erkannte Werte	40,63%
Erkannte Farben	59,38%
Insgesamt erkannt	50,00%
Insgesamt korrekt erkannte Karten	28,83%

Abbildung 15: Erkennungsraten bei verändertem Blickwinkel

7 Fazit / mögliche Erweiterungen

7.1 Fazit

Es hat sich gezeigt, dass die Berechnung von Wahrscheinlichkeiten zu verschiedenen Spielphasen beim Texas Hold'Em nicht so einfach ist, wie man denken könnte. Es gibt zwar mathematische Ansätze, die die Wahrscheinlichkeit eine Hand zu erhalten berechnen, diese sind jedoch sehr allgemein gehalten. Einen umsetzbaren Ansatz zu finden, der zu verschiedenen Phasen des Spiels aus gegebenen Karten Wahrscheinlichkeiten generiert, gestaltete sich als schwierig und besonders Zeitintensiv.
Auch die Wahl eines geeigneten Verfahren zur Erkennung der Spielkarten als Marker konnte erst nach mehreren Ansätzen erfolgen. Erste Versuche, die Karten über das vorgestellte SIFT-Verfahren zu erkennen, schlugen fehl. Die extrahierten Merkmale der Karten waren nicht eindeutig genug. Außerdem lief die Erkennung sehr langsam, was keine Echtzeit-Fähigkeit des Systems zugelassen hätte. Auch andere, ähnlich komplexe Ansätze, ergaben keine zufriedenstellenden Ergebnisse. Im Endeffekt stellte sich die Verwendung des relativ simplen Template Matchings als richtige Wahl heraus. Bei guten Lichtverhältnissen und einem fest montierten Kamera lieferte das Verfahren absolut zuverlässige Erkennungsraten. Bei Aufnahmen aus der Hand stößt das System jedoch an seine Grenzen. Auch die Art und vor allem Auflösung der verwendeten Kamera haben starken Einfluss auf die Stabilität der Erkennung.

7.2 Mögliche Erweiterungen

Die entwickelte Applikation kann dem Spieler Auskünfte über seine Hand geben, die als Grundlage für die weiteren Handlungen dienen können. In der Praxis, also bei einem echten Pokerspiel, ist eine sinnvoll Verwendet jedoch nicht möglich. Eine Weiterentwicklung in diese Richtung wäre daher der logische nächste Schritt. Da das System nicht sehr invariant gegenüber Veränderungen der Kameraposition ist, müsste ein Ansatz gefunden werden, der Karten aus allen möglichen Blickwinkeln korrekt erkennt. Dann wären verschiedene Erweiterungen denkbar, die auch einen Einsatz des System in der Praxis ermöglichen würden. Es wäre zum Beispiel denkbar, dass der Spieler ein HMD (Head-Mounted Display) trägt. Dieses könnte die Karten aus dem Blickwinkel des Spielers einlesen und ihm im Disyplay des HMDs die Wahrscheinlichkeiten anzeigen. Dies würde auch die Erweiterung in Richtung einer richtigen AR-Anwendung, mit virtuellen Objekten die in die Umgebung eingeblendet werden, interessanter machen. Im Bezug auf das Pokerspiel wäre es Möglich, dem Spieler noch weitere Informationen zu bieten. Vor allem Aussagen darüber, ob ein Spieler mit seinen Karten mitbieten, seinen Wetteinsatz erhöhen oder aussteigen soll, wären sicherlich hilfreich.

Abbildungsverzeichnis

8 Literaturverzeichnis

Literatur

[Zheng2007] Zheng, C. Dr. Green, R., New Zealand, 2007. Playing Card Recognition Using Rotational Invariant Template Matching. Stand: 25. Januar 2010. http://digital.liby.waikatohttp//digital.liby.waikato.ac.nz/conferences/ivcnz07/papers/ivcnz07-paper51.pdf.ac.nz/conferences/ivcnz07/papers/ivcnz07-paper51.pdf

[Brad2008] Bradley, J., Mai 2008. Webcam Based Online Poker Trainer. Stand: 25. Januar 2010. http://www.cs.unm.edu/~drwombat/Vision_final_project.pdf

[Cole2004] Cole, L., Austin, D., Australia 2004. Visual Object Recognition using Template Matching. Stand: 25. Januar 2010. http://www.araa.asn.au/acra/acra2004/papers/cole.pdf

[Lowe2004] Lowe, D. G., Canada 2004. Distinctive Image Features from Scale-Invariant Keypoints. Computer Science Department, University of British Columbia. Stand: 25. Januar 2010. http://people.cs.ubc.ca/~lowe/papers/ijcv04.pdf

[Wuest2006] Wuest, H., 2006. Bildbasiertes Tracking. Vorlesung, Fraunhofer IDG. Stand: 25. Januar 2010. http://www.igd.fhg.de/~hwuest/vorlesung06/BildbasiertesTracking.pdf

[opencv2008] Bradski, G., Kaehler, A., USA 2008. Lerning OpenCV. O'Reilly

[Erhardt] Erhardt, A., Einführung in die Digitale Bildverarbeitung: Grundlagen, Systeme und Anwendungen, Vieweg+Teubner

[Gonz2002] Gonzales, R., Woods, R., Digital Image Processing, Prentice Hall India, 2002.

[Farb2005] Färber, M, 2005. Markerbasiertes Tracking für Augmented Reality Applikationen. Stand: 25. Januar 2010. http://www.vs.inf.ethz.ch/edu/SS2005/DS/reports/03.2-ar-report.pdf

[Brauer2008] Brauer, H., 2008. Augmented Reality Tracking. Seminararbeit, Universität Hamburg. Stand: 25. Januar 2010. http://users.informatik.haw-hamburg.de/~ubicomp/projekte/master08-09-aw/brauer/bericht.pdf

[Maiero2009] Maiero, J., 2009. Image-based Tracking, Masterarbeit, FH Bonn. Stand: 25. Januar 2010. http://cg.inf.fh-bonn-rhein-sieg.de/wp-content/uploads/2009/10/imagebased_tracking_jm09.pdf

[Schneider2007] Schneider, M. 2007. Markerloses Posentracking für Augmented Reality Anwendungen, Universität Ulm. Stand: 25. Januar 2010. http://beached.org/static/files/markerless_pose_tracking.pdf

[Lewis1995] Lewis, J. P., Kanada 1995.bFast TemplateMatching,Canadian Image Processing and Pattern Recognition Society. Stand: 25. Januar 2010. http://scribblethink.org/Work/nvisionInterface/vi95_lewis.pdf

[Lod2004] Lodermann, C., Lambers, M., 2004. Objekterkennung in Bilddaten, Seminar: „Unterstützung von Landminendetektion durch Bildauswertungsverfahren und Robotereinsatz", Universität Münster. Stand: 25. Januar 2010.

[Meyer2008] Meyer, L., Görge, T., 2008. Opponent Modeling im Poker, Seminararbeit, Universität Darmstadt. Stand: 25. Januar 2010. Stand: 25. Januar 2010. http://www.ke.tu-darmstadt.de/lehre/archiv/ss08/challenge/Ausarbeitungen/Meyer_Goerge.pdf

[Straßer2003] Straßer, C., 2003. Kantendetektion in der Bildverarbeitung, Seminararbeit, Universität Passau. Stand: 25. Januar 2010. Stand:

25. Januar 2010. http://students.fim.uni-passau.de/lehrstuehle/donner/lehrveranstaltungen/seminarWS02/vortrag05.pdf

[Als2000] Alspach, B., 2000. 7-Card Poker Hands. Stand: 25. Januar 2010. http://www.math.sfu.ca/~alspach/comp20/

[QA] QA - Poker Probabilities. Stand: 25. Januar 2010. http://www.probabilityof.com/poker.shtml