H. Wandke, S. Kain & D. Struve (Hrsg.): Mensch & Computer 2009: Grenzenlos frei!?
München: Oldenbourg Verlag, 2009, S. 243-252

Gestenerkennung für Multitouch-Anwendungen in UI-Frameworks

Peter Barth, Thomas Leidecker

Medieninformatik, Hochschule RheinMain

Zusammenfassung

Gestenbasierte Multitouch-Anwendungen erfreuen sich zunehmender Beliebtheit, werden jedoch meist systemspezifisch mit hohem Aufwand realisiert, da standardisierte Frameworks fehlen. Basierend auf der Analyse typischer Gesten, die Tippen, Bewegen, Distanzänderungen und Drehen beinhalten, wird eine Architektur vorgeschlagen, um Gestenerkenner einfach zu realisieren. Objekte können mit Gestenerkennern assoziiert werden und Anwendungsentwickler werden – wie von klassischen UI-Frameworks gewohnt – über entsprechende Ereignisse informiert. Framework und Demonstrationsanwendung wurden auf einem aktuellen Multitouch-Tisch mit Flash als Umgebung realisiert.

1 Einleitung

Multitouch-Systeme bieten eine intuitive und attraktive Alternative zu Systemen, die auf herkömmlichen Eingabegeräten wie Maus oder Tastatur basieren und bei denen nur eine indirekte Interaktion mit der Benutzungsoberfläche möglich ist. Mit einer einzelnen Geste kann das ausgedrückt werden, was mit Maus und Tastatur nur in mehreren Schritten möglich ist (Moscovich 2007). Außerdem können auch mehrere Personen gleichzeitig und gemeinsam in einer Umgebung arbeiten (Muto & Diefenbach 2008). Technische Innovationen ermöglichen Multitouch-Displays auch im Massenmarkt (Apple 2005). Selbst größere Multitouch-Displays, wie in Abbildung 1, verbreiten sich immer mehr. Während herkömmliche Eingabegeräte und darauf basierende Interaktionen standardisiert und in grafische Toolkits integriert sind, ist dies für Multitouch-Interaktionen nicht der Fall. Die Erstellung von Multitouch-Anwendungen ist nicht so komfortabel, wie die Erstellung klassischer Anwendungen und das Ergebnis ist meist spezifisch für eine bestimmte Hardware. Der Tisch aus Abbildung 1 liefert zum Beispiel nur einen Datenstrom mit erkannten Berührpunkten und stellt weder Gestenerkennung noch Integration in ein UI-Framework zur Verfügung.

In diesem Beitrag stellen wir eine Architektur und Implementierung für die Erkennung von Gesten und deren Integration in ein UI-Framework vor. Wir abstrahieren von konkreter Hardware und verwenden als Eingabe einen Strom von Berührpunkten. Ausgehend von einer

systematischen Untersuchung typischer Gesten (Wu & Balakrishnan 2003) extrahieren wir für die Erkennung notwendige Informationen. Darauf aufbauend schlagen wir für ausgesuchte Gesten Erkennungsverfahren vor. Wir präsentieren dann eine Architektur, mit der An-

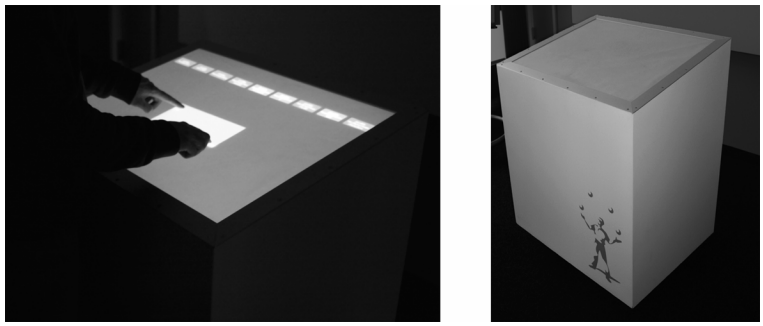


Abbildung 1: Multitouch-Tisch, mediaman

wendungsentwickler schnell Multitouch-Anwendungen erstellen können und gegebenenfalls eigene Gestenerkenner integrieren können. Ein entsprechendes Framework wurde in Flash realisiert und darauf aufbauend Beispielanwendungen entwickelt.

2 Multitouch-Umgebungen

Der Light Pen von Sketchpad (Sutherland 1963) war das erste direkte Eingabegerät, wurde jedoch bald aus Kostengründen durch die Maus ersetzt. Sie ist heute das meist verwendete Zeigegerät (Schedlbauer 2007). Mit dem Touchscreen wurde es ab 1977 wieder möglich direkt mit dem Computer zu interagieren und seitdem wird auch aktiv an Möglichkeiten gearbeitet mehrere Finger zu verwenden (Schöning et al. 2008). Durch kostengünstige Varianten von Multitouch-Tischen (Han 2005) wurde die Multitouch-Technologie auf großen Displays für den Massenmarkt tauglich. Neben der Möglichkeit mehrere Finger zur Eingabe zu verwenden, macht bei Touch-Umgebungen die Kombination von Ein- und Ausgabe die direkte Interaktion möglich (Buxton 2005). Der von der Firma *mediaman* zur Verfügung gestellte Multitouch-Tisch aus Abbildung 1 basiert auf der *verhinderter Totalreflexion* (Han 2005). Ein Finger muss die Oberfläche berühren, um einen Effekt zu erzielen. Das System verfügt über eine Erkennungssoftware, welche die Berührpunkte von Fingern mit der Tischoberfläche erkennt und mit fester Rate einen Datenstrom (Kaltenbrunner 2005) weiterleitet. Die Interaktionsoberfläche wird mit Hilfe von Rückprojektion auch als Ausgabemedium verwendet. Wir gehen im Folgenden davon aus, Berührpunkte mit entsprechender Identifikation als absolute Koordinaten in Bildschirmauflösung zu erhalten.

Wu und Balakrishnan geben eine systematische Untersuchung von Gesten für eine Multitouch-Umgebung (Wu & Balakrishnan 2003). Es wird klar, dass bei der Gestenerkennung Startzeitpunkt, dynamische Phase und Endzeitpunkt identifiziert werden müssen. Die im

Gegensatz zur Mauspositionierung geringere Genauigkeit sollte bei der Gestenerkennung berücksichtigt werden. Es wird außerdem zwischen einer Geste und deren Bedeutung unterschieden und vorgesehen, dass eine Geste je nach Kontext unterschiedliche Bedeutung haben kann. Einen Überblick über verschiedene Ansätze zur Gestenerkennung und deren Integration in ein UI-Framework gibt (Echtler & Klinker 2008). Es wird eine Architektur ähnlich zu der hier vorgestellten vorgeschlagen. Die Gestenerkennung wird in einem *Interpretation Layer* behandelt und es ist möglich Gestenerkennung auf Regionen zu beschränken.

3 Multitouch-Gesten

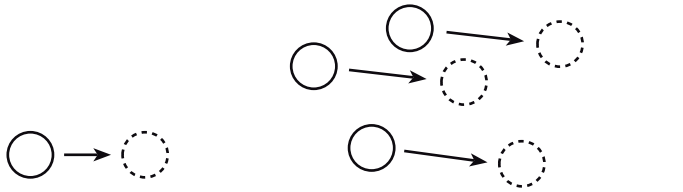
Gesten ermöglichen die Interaktion mit einer Multitouch-Anwendung. Mit ihnen können Objekte selektiert und manipuliert werden. Im Folgenden analysieren wir mehrere Multitouch-Gesten und identifizieren Eigenschaften, die sowohl für die Erkennung als auch die Ereignisse in einem UI-Framework relevant sind.

(a) Doppelklick *(b) Zeitklick* *(c) Zweifingerklick*

Abbildung 2: Klicken

Bei der *Doppelklick*-Geste in Abbildung 2(a) wird das Verhalten eines Maus-Doppelklicks nachgeahmt. Dazu wird zweimal kurz hintereinander dasselbe Objekt mit einem Finger berührt. Bei der Erkennung dieser Geste muss innerhalb zeitlicher und räumlicher Toleranzen erkannt werden, ob derselbe Finger die Oberfläche zweimal berührt hat. Die *Zeitklick*-Geste in Abbildung 2(b) wird ausgeführt, indem man mit einem Finger für eine gewisse Zeitspanne auf einem Objekt bleibt und sich der Finger innerhalb dieser Zeit, im Rahmen von Toleranzen, nicht bewegt. Bei dem *Zweifingerklick* in Abbildung 2(c) berühren zwei Finger annähernd gleichzeitig ein Objekt und verlassen es auch nahezu simultan wieder. Werden während der Berührung mehr als zwei Finger verwendet, wird die Geste richtigerweise nicht erkannt.

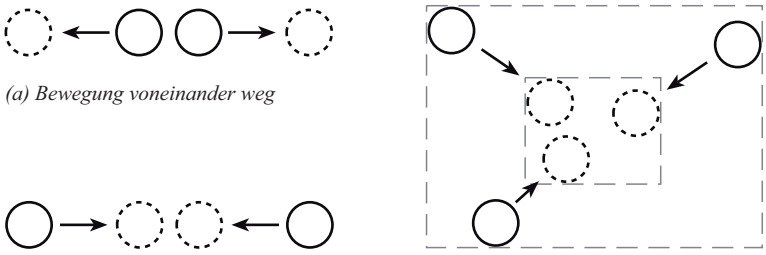
Bei der Geste *Bewegen eines Fingers* in Abbildung 3(a) berührt genau ein Finger ein Objekt und bewegt sich beliebig auf der Interaktionsoberfläche, bis er diese wieder verlässt. Eine typische Standardinterpretation dieser Geste ist, dass das Objekt mit der ersten Berührung ausgewählt und um denselben Bewegungsvektor wie der zugeordnete Finger bewegt wird. Bei der Geste *Bewegung mehrerer Finger in eine Richtung* in Abbildung 3(b) bewegen sich mindestens zwei bis maximal fünf Finger in eine Richtung. Das Objekt wird dabei standardmäßig um den mittleren Bewegungsvektor der zugeordneten Finger bewegt. Der Schwerpunkt der erkannten Finger eignet sich dazu leider nicht, da sich dieser bei Hinzunahme eines Fingers sprunghaft verschieben würde. Der mittlere Bewegungsvektor ist also toleranter und intuitiver sowohl in der Erkennung als auch in der Verwendung.



(a) Bewegen eines Fingers *(b) Bewegen mehrerer Finger in eine Richtung*

Abbildung 3: Bewegen der Finger

Bei der Geste *Distanzänderung zweier Finger* in Abbildung 4(a) und (b) bewegen sich zwei Finger aufeinander zu oder voneinander weg. Dies entspricht der typischen Verwendung zweier Zeigefinger zum Skalieren. Der Skalierungsfaktor wird durch die Längenänderung der Strecke zwischen den Fingern ermittelt. Bei der *Distanzänderung mehrerer Finger* in Abbildung 4(c) bewegen sich mindestens drei und maximal fünf Finger aufeinander zu oder voneinander weg. Dies entspricht einer Greif- bzw. Spreizbewegung der Finger einer Hand zum Skalieren. Der Skalierungsfaktor wird durch die Größenänderung der umschließenden Bounding Box bestimmt.

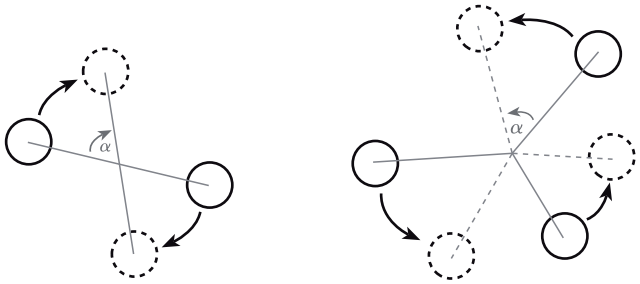


(a) Bewegung voneinander weg

(b) Bewegung aufeinander zu *(c) Distanzänderung mehrerer Finger*

Abbildung 4: Distanzänderung der Finger

Bei der Geste *Kreisbewegung mit zwei Fingern* in Abbildung 5(a) bewegen sich zwei Finger um ihren Mittelpunkt. Bei der *Kreisbewegung mit mehreren Fingern* in Abbildung 5(b) drehen sich zwischen drei und fünf Finger um ihr Zentrum. Eine übliche Interpretation beider Gesten ist die Drehung des selektierten Objekts um den Schwerpunkt der Berührpunkte.



(a) Kreisbewegung mit zwei Fingern *(b) Kreisbewegung mit mehreren Fingern*

Abbildung 5: Kreisbewegung der Finger

Eine Gestenbewegung muss aber nicht immer dieselbe Reaktion hervorrufen, sondern kann je nach Situation und Kontext eine andere Bedeutung haben. So muss eine Kreisbewegung mit zwei Fingern nicht zwangsläufig das Rotieren des betroffenen Objektes bedeuten, sondern könnte auch als Farbänderung interpretiert werden. Es ist also sinnvoll die Gestenbewegung von deren Interpretation zu trennen. Trotzdem kann häufig eine typische Umsetzung für eine Geste, wie zum Beispiel Selektion bei Berührung angenommen werden.

4 Gestenerkennung

Während einer Gestenerkennung werden Berührpunkte, welche die absolute Position zu einem gewissen Zeitpunkt enthalten, kontinuierlich analysiert. Zuerst wird jeder Berührpunkt einem Multitouch-Objekt zugeordnet – er wird *registriert*. Basierend auf den registrierten Berührpunkten können Schwerpunkt, Bounding Box, Bewegungsvektor und Rotationswinkel abgeleitet werden. Mit Hilfe dieser Informationen können alle vorgestellten Gesten kontextunabhängig erkannt werden. Ein Gestenerkenner kombiniert nun diese Informationen um Startzeitpunkt, dynamische Phase und Endzeitpunkt zu bestimmen. Dabei werden Ungenauigkeiten der Erkennungssoftware oder ungewollte, kleine Bewegungen der Finger gestenspezifisch durch zeitliche und räumliche Toleranzen kompensiert. Im Folgenden stellen wir Besonderheiten der Erkennung ausgesuchter Gesten vor und spielen die Erkennung einer Geste vollständig durch. Details zu allen Gesten sind in (Leidecker 2009) aufgeführt.

Bei vielen Gesten wird eine sofortige Reaktion von den Interaktionselementen erwartet, wie bei der *Kreisbewegung mit zwei Fingern*, bei der sich das betroffene Objekt in Echtzeit mitdrehen soll. Berührpunktdaten zu sammeln und anschließend zu analysieren, ist in diesem Fall nicht sinnvoll. Für eine Geste muss es Startzeitpunkt, Endzeitpunkt und dynamische Phase geben. Die Anzahl der Berührpunkte ist ein entscheidendes Attribut für die Erkennung einer Geste. So wird beispielsweise die dynamische Phase erst gestartet, wenn die Anzahl der Berührpunkte mit der Definition der Geste übereinstimmt. Auch wird bei vielen Gesten die Gestenerkennung ausgesetzt, wenn die Anzahl der Berührpunkte von der Definition abweicht. Während bei der Anzahl der Berührpunkte keine Toleranzen notwendig sind, funktioniert zum Beispiel der *Zeitklick* nur mit räumlichen Toleranzen stabil. Beim Zeitklick soll

der Finger für eine gewisse Zeit ruhen. Eine Bewegungstoleranz sorgt dafür, dass ungewollte, kleine Bewegungen abgefangen werden, die schon durch Druckpunktverlagerung der Fingerspitze entstehen oder durch Ungenauigkeiten der Erkennungssoftware hervorgerufen werden. Bei dem *Zweifingerklick* in Abbildung 6 sind sowohl die Anzahl der Berührpunkte als auch die Verwendung einer Zeittoleranz für die Gestenerkennung entscheidend. Der Zweifingerklick wird erkannt, wenn zwei Finger das Objekt zeitgleich oder in einem gewissen Zeitrahmen berühren und wieder verlassen. Sind zu einem Zeitpunkt mehr als zwei Finger auf dem Objekt, wird die Geste deaktiviert. Sie wird erst wieder aktiviert, wenn alle Finger das Objekt verlassen, wodurch ein ungewolltes Auslösen der Geste verhindert wird.

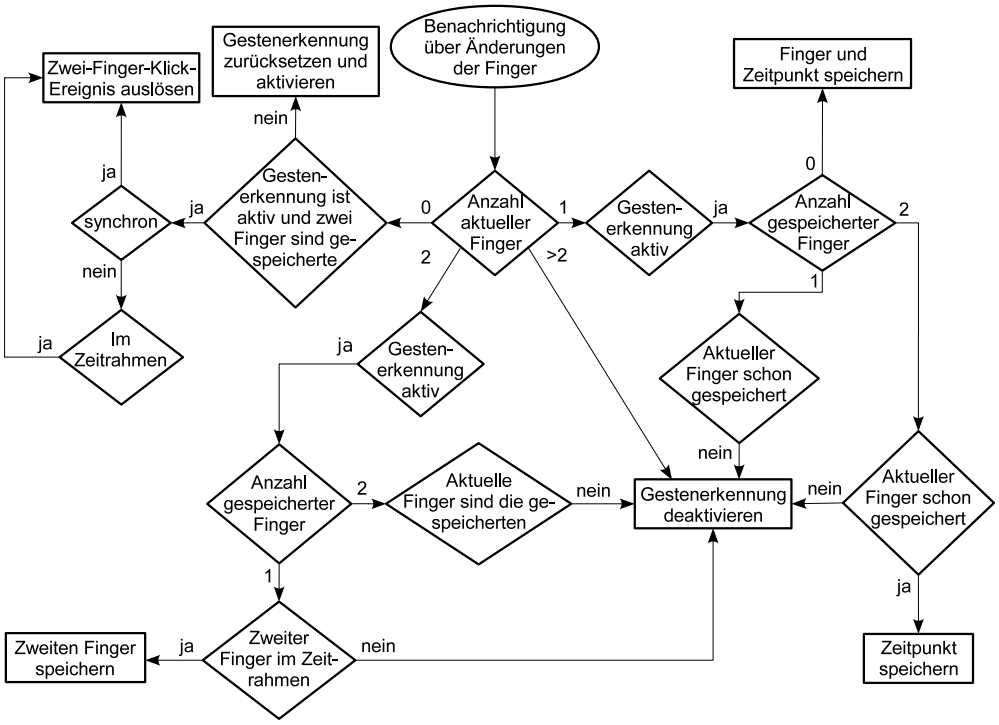


Abbildung 6: Ablauf Zweifingerklick

Wir betrachten nun im Detail die Erkennung des Zweifingerklicks in Abbildung 6 (nicht weiter verfolgte Pfade im Ablauf lösen keine weiteren Aktionen aus). Das Erkennungsverfahren muss zeitliche Verzögerungen zwischen den Berührungen berücksichtigen. Dazu werden Berührpunkte mit Fingern assoziiert. Wird ein Berührpunkt erstmals für ein Objekt registriert, wird er als erster Finger identifiziert. Anhand des Zeitpunkts des ersten Fingers wird später entschieden, ob ein weiterer Berührpunkt der nötige zweite Finger ist, der innerhalb einer kurzen Verzögerung hinzugekommen ist. Wurde bereits ein Berührpunkt registriert, muss dies der aktuelle Finger auf dem Objekt sein. Falls nicht, ist das seltene Ereignis aufgetreten, dass der Finger entfernt wurde und zeitgleich ein neuer Finger auf dasselbe

Objekt gesetzt wurde. Da dies nicht dem gewünschten Verhalten entspricht, wird die Geste vorläufig deaktiviert. Waren schon zwei Finger registriert und ist jetzt nur noch ein einziger Finger vorhanden, dann muss dieser verbleibende Finger innerhalb einer kurzen Verzögerung das Objekt verlassen, um das Ereignis der Geste auszulösen. Werden gleichzeitig zwei Berührpunkte für ein bisher unberührtes Objekt registriert, sind beide Finger zeitgleich hinzugekommen. Wenn bereits zwei Berührpunkte registriert sind, dürfen diese in der dynamischen Phase nicht durch andere ersetzt werden. Berührt bei aktiver Geste kein Finger mehr das Objekt, nachdem zuvor zwei Finger auf dem Objekt waren, dann wurden beide Finger entfernt und ein Ereignis für die Geste wird ausgelöst. Die Gestenerkennung wird bei Fehlern, wie einer falschen Anzahl an Fingern oder ein Überschreiten der Zeittoleranzen, temporär deaktiviert, bis kein Berührpunkt mehr für das Objekt registriert ist. Nach Auslösen des Ereignisses wird der Gestenerkenner in eine aktive Ausgangsposition zurückgesetzt. Im Ereignis der Geste werden gestenspezifische Informationen, wie zum Beispiel der Mittelpunkt zwischen den beiden Fingern beim Zweifingerklick oder der Drehwinkel bei der Kreisbewegung mit zwei Fingern, mit dem Ereignis verknüpft. Ein Problem mancher Gesten ist, dass man auf einen Finger reagieren möchte, der die Interaktionsoberfläche zeitweise verlässt, wie beispielsweise beim *Doppelklick*. Aufgrund der Multitouch-Fähigkeit müssen im Gegensatz zum Maus-Doppelklick mehrere, gleichzeitige Klicks unterschieden werden. Verlässt ein Finger die Oberfläche und berührt sie danach erneut, ist dies für die Erkennungssoftware ein neuer Berührpunkt. Dass dieser von demselben Finger hervorgerufen wurde, kann nicht erkannt werden. Um einen Zusammenhang zwischen den beiden Klicks herzustellen, muss der zweite nicht nur in einer gewissen Zeitspanne erfolgen, sondern auch innerhalb eines räumlichen Toleranzbereichs um den ersten Klick liegen.

5 Architektur und Umsetzung

Das Framework in Abbildung 7 setzt auf den Daten der Erkennungssoftware auf. Die Berührpunkte, welche durch die Berührung des Fingers mit der Interaktionsoberfläche entstehen, werden von der Erkennungssoftware erfasst und ihre Position und Identifikation an das Framework übermittelt. Ein *Datenempfänger* nimmt diese Daten entgegen, extrahiert die Informationen der Berührpunkte und bereitet diese Daten für die Weiterverarbeitung auf. Die Punkte werden in das Koordinatensystem der Anwendung überführt und falls nötig kalibriert. Bei Bedarf kann diese Schicht ausgetauscht oder angepasst werden. Die aufbereiteten Daten werden an die nächst höhere Verarbeitungsschicht des Frameworks, den *Datenverwalter*, weitergegeben. Der Datenverwalter dient als Schaltzentrale innerhalb des Frameworks. Er interpretiert die Daten und stößt weitere Verarbeitungsschritte an. Die Daten werden als Berührpunkte interpretiert und gespeichert, aktualisiert oder gelöscht. Der *Berührpunktverwalter* organisiert alle Berührpunkte und kann Auskunft über einzelne Berührpunkte, die Gesamtheit der Berührpunkte oder Mengen bestimmter Berührpunkte geben. Er berechnet beispielsweise den Schwerpunkt von Berührpunkten oder ob sich mehrere Berührpunkte in die gleiche Richtung bewegen. Diese Informationen werden von jedem *Gestenerkenner* benötigt, welcher die Verhaltensweisen der Berührpunkte analysieren, um festzustellen, ob eine Geste eingetreten ist und um deren Eigenschaften zu bestimmen. Werden Gesten erkannt,

sollen die Objekte der Anwendungsschicht darauf reagieren können. Gestenerkenner werden einem *Multitouch-Objekt* zugeordnet. Die Multitouch-Objekte werden in einem UI-Framework mit einer grafischen Darstellung assoziiert und kapseln selbst nur Multitouch-Zustand und Verhalten. Ein Multitouch-Objekt hält eine Liste von Gestenerkennern, wodurch dynamisch beliebig viele Gesten erkannt werden können. Es besitzt außerdem einen lokalen Berührpunktverwalter, der nur ihm zugewiesene Berührpunkte registriert. Der Gestenerkenner kann so schneller arbeiten, da er nur für das Objekt relevante Berührpunkte betrachten muss.

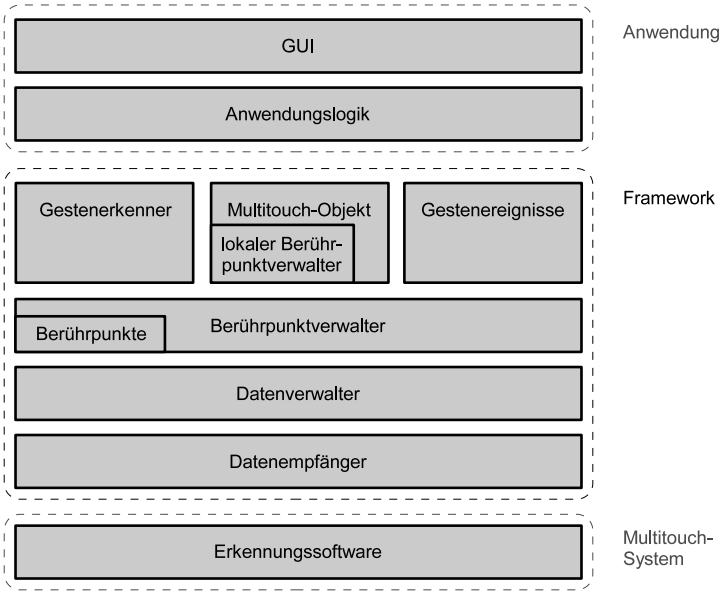


Abbildung 7: Komponenten des Frameworks

Damit in der Anwendungsschicht individuell auf Gesten reagiert werden kann, lösen die Gestenerkenner zwischen erkannten Startzeitpunkt und Endzeitpunkt *Gestenereignisse* aus. Diese Ereignisse beinhalten gestenspezifische Informationen als Attribute, die in der Anwendungsschicht verwendet werden können. Das Framework ist leicht um weitere Gesten erweiterbar. Die Gestenerkenner sind entkoppelt vom Kern des Frameworks und können in einer eigenen Klasse separat definiert werden. Der Zugriff auf die Hilfsfunktionen des Berührpunktverwalters zur Berechnung abgeleiteter Informationen erleichtert dabei die Realisierung des Gestenerkenners. Das Framework ruft betroffene Gestenerkenner bei Änderung relevanter Berührpunkte auf und die Anwendungsschicht erwartet nur noch die Ereignisse der Gesten.

Zur Realisierung des Frameworks wurde Flash mit Actionscript 3.0 gewählt. Durch einen integrierten Umgang mit Grafikobjekten und einem anpassbaren Ereignismodell eignete sich Flash gut für die Umsetzung eines Frameworks, das im Bereich der grafischen Benutzungsschnittstellen einzuordnen ist. Viele der prototypischen Multitouch-Anwendungen werden

zurzeit in Flash realisiert. Das Anwendungsbeispiel *Foto-Betrachter* in Abbildung 8 demonstriert die Funktionalität des Frameworks. Mit dieser Multitouch-Anwendung können Fotos betrachtet und manipuliert werden – auch mit mehreren Personen. Auf der linken Seite befinden sich Funktionen und Gesten, die auf Fotos angewendet werden können. Die Fotos können mit einem oder mehr Fingern verschoben, skaliert oder rotiert werden. Da das Framework mehrere Fokusse ermöglicht, sind diese Aktionen mit verschiedenen Fotos gleichzeitig möglich. Zur Laufzeit werden Gestenerkenner dynamisch mit Multitouch-Objekten verbunden und jeweils mit unterschiedlichem Verhalten assoziiert.

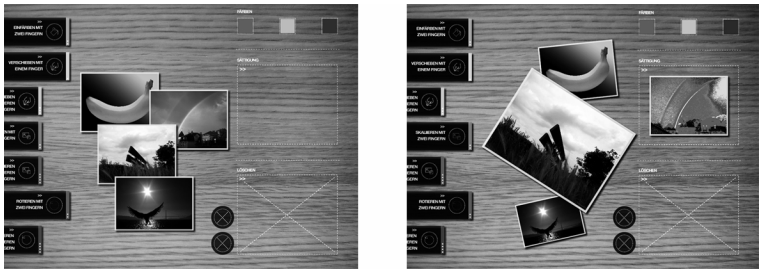


Abbildung 8: Anwendung Foto-Betrachter

Klickt man mit zwei Fingern auf ein Foto während man mit einem anderen Finger ein Farbfeld berührt, so kann man dieses einfärben. Die Farbsättigung eines Fotos lässt sich anpassen, indem man dieses in einen speziellen Arbeitsbereich verschiebt. Mittels einer Kreisbewegung zweier Finger, die vorher noch zum Drehen des Fotos gedient hat, wird in diesem Arbeitsbereich die Sättigung angepasst. Dies ist ein Beispiel für einen sinnvollen Einsatz der Trennung von Gestenbewegung und deren Interpretation.

6 Zusammenfassung und Ausblick

Typische Gesten in Multitouch-Umgebungen können mit wenigen Informationen wie Schwerpunkt, Bounding Box, Bewegungsvektor und Rotationswinkel erkannt werden. Gestenerkenner kombinieren diese Informationen und berücksichtigen dabei zeitliche sowie räumliche Toleranzen. Darauf aufbauend präsentieren wir eine Architektur zur Gestenerkennung und Integration in ein UI-Framework. Die vorgeschlagene Architektur eignet sich gut zur Realisierung eigener Gesten und Gestenerkenner. Die Flash-basierte Implementierung und Realisierung der Demonstrationsanwendung lässt Anwendungsentwickler schnell umfangreiche Multitouch-Anwendungen erstellen. Zurzeit werden mit Hilfe des Frameworks Anwendungsszenarien realisiert und dabei weitere Gestenerkenner entwickelt. Ob die abgeleiteten Informationen ein Großteil noch zu realisierender Gestenerkenner ausreichend unterstützt, muss noch evaluiert werden. Für einen produktiven Einsatz muss die Kalibrierung der Umgebung sowie die Assoziation der Finger zu Berührpunkten verbessert werden.

Danksagung

Wir danken *mediaman* für die Unterstützung und Bereitstellung des Multitouch-Tisches.

Literaturverzeichnis

Apple (2005). *Multipoint Touchscreen*. Schutzrecht WO 2005/114369 A2, G06F 3/033, US, Apple.

Buxton,W. (2005). *Taxonomies Of Input*. In *Haptic Input*.

Echtler, J. & Klinker, G. (2008). A Multitouch Software Architecture. In *NordiCHI 2008: Using Bridges*. Technische Universität München.

Han, J. Y. (2005). Low-cost multi-touch sensing through frustrated total internal reflection. In *UIST '05: User interface software and technology*. New York: ACM, S. 115–118.

Leidecker, T. (2009). *Framework zur Entwicklung interaktionsgesteuerter Applikationen für einen Multitouch-Tisch*, Diplomarbeit, Fachhochschule Wiesbaden.

Kaltenbrunner, M. et al. (2005). Tuio - a protocol for table based tangible user interfaces. In *6th Workshop on Gesture in Human-Computer Interaction and Simulation*. Vannes.

Moscovich, T. (2007). *Principles and applications of multi-touch interaction*. PhD thesis, Brown University.

Muto, W. & Diefenbach, P. (2008). Applications of multi-touch gaming technology to middle-school education, In *ACM SIGGRAPH 2008 posters*. New York: ACM.

Schedlbauer, M. J. (2007). *A Survey of Manual Input Devices*. Technical report, University of Massachusetts.

Schöning, J. et al. (2008). *Multi-Touch Surfaces: A Technical Guide*. Technical University of Munich.

Sutherland, I. E. (1963). *Sketchpad: a man-machine graphical communication system*. In *AFIPS '63*. New York: ACM, S. 329-346.

Wu, M. & Balakrishnan, R. (2003). Multi-finger and whole hand gestural interaction techniques for multi-user tabletop displays. In *UIST '03, User interface software and technology*. New York: ACM, S. 193-202.

Kontaktinformationen

Peter Barth (barth@informatik.fh-wiesbaden.de)
Thomas Leidecker (thomas.leidecker@gmx.de)