Prof. Dr. Michael Rohs, Dipl.-Inform. Sven Kratz

Mensch-Maschine-Interaktion 2 (Mobile Interaktion) WS 2011/2012

Übungsblatt 4

Aufgabe 1: Gesture Wizards [Einzelabgabe]

In der Vorlesung wurde Template-basierte Gestenerkennung für Touch-Screens vorgestellt. In Android ist bereits ein Template-basierter Gestenerkenner eingebaut, der für diese Übung verwendet werden soll. Zwei Werkzeuge zur Erstellung von Gesten können von der Webseite heruntergeladen werden. Mit dem „GestureBuilder“ können Touch-Screen-Gesten eingegeben und auf der SD-Karte abgespeichert werden. Andere Anwendungen können diese Gesten dann lesen. „GestureDemo“ ist ein einfaches Programm, mit dem Gesten getestet werden können. Es greift auf die mit GestureBuilder erstellten Gesten zurück. Bei Eingabe einer Geste wird das am besten passende Template gesucht und ausgegeben. Beide Werkzeuge liegen im Quelltext vor.

In dieser Aufgabe geht es darum, verschiedene Arten von Animationen durch Gesteneingabe zu steuern. Das Szenario lautet wie folgt: Zwei Zauberer stehen sich auf Berggipfeln gegenüber und tragen ein magisches Duell aus (siehe Abbildung). Die Zauberer können offensive und defensive Zaubersprüche von sich geben (ausgelöst durch Gesteneingabe). Um den anderen Zauberer anzugreifen muss der Zauberer einen offensiven Zauberspruch sagen. Offensive Zauber („air“, „earth“, „fire“) wandern über eine gewisse Zeit (4s) vom einen zum anderen Zauberer. Ankommende offensive Zaubersprüche müssen durch defensive Zaubersprüche des selben Typs abgewehrt werden. So kann z.B. ein ein „fire“ Angriff durch „deffire“ abgewehrt werden. Defensive Zaubersprüche („defair“, „defearth“, „deffire“) bauen eine schützende Aura um den Zauberer herum auf. Initial hat jeder Zauberer 10 Punkte. Nicht abgewehrte Zauber verursachen einen Schadenspunkt. Wer zuerst 0 Punkte erreicht, hat verloren. Ein Zauberer kann zu einem Zeitpunkt immer nur maximal eine schützende Aura haben, die für eine begrenzte Zeit (5s) aktiv bleibt. Durch einen offensiven Zauberspruch desselben Zauberers wird die schützende Aura zerstört. Folgende 6 Zaubersprüche existieren also: 3 offensive: „air“, „earth“, „fire“ und 3 defensive: „defair“, „defearth“, „deffire“. Für jeden Zauberspruch wird eine Geste benötigt.

Die Einhaltung der Spielregeln und der Spielablauf liegen als Gerüst vor. Sie können das Gerüst von der Webseite der Vorlesung herunterladen. Das Codegerüst verwendet Android 2.3.3, so dass es auf aktuellen Geräten ausführbar ist. Schauen Sie sich die Kommentare im Quelltext an und machen Sie sich mit dem Codegerüst vertraut. Die Teilaufgaben lauten:

a) Erstellung von Bitmaps des Hintergrundbildschirms (800x480px), der beiden Zauberer (ca. 144x226px), der drei offensiven (ca. 100x100px) und der drei defensiven (ca. 240x240px) Zaubersprüche. In den Ressourcen sind diese Elemente bereits beispielhaft als png-Dateien gespeichert. Die von Ihnen erstellten Objekte müssen die angegebenen Größen haben.
b) Benutzer-definierte Gesten: In Gesten-basierten Systemen hat sich gezeigt, dass verschiedene Benutzer oft verschiedene Gesten für die gleiche Aktion bevorzugen. Oft definieren Designer Gesten, die Endbenutzer wenig intuitiv finden. Eine Alternative besteht darin, das Gesten-Set von Benutzern definieren zu lassen. D.h. die Benutzer

bekommen den Effekt der Geste erklärt (z.B. „Air"-Zauberspruch im Kontext des Spiels) und sollen sich eine Geste dazu ausdenken. Dann wird geschaut, wie groß die Übereinstimmung zwischen verschiedenen Benutzern ist und die Geste verwendet, die von den meisten Benutzern vorgeschlagen wurde. Verfahren Sie nach diesem Ansatz in der Entwicklung des Gesten-Sets für das Spiel. Erklären Sie unabhängig voneinander 3 Benutzern das Spiel (auch an Hand der Abbildung unten) und bitten Sie die Benutzer, die Geste für jeden Zauberspruch auf ein Blatt Papier zu zeichnen. Konsolidieren Sie basierend darauf die Gesten und entwickeln Sie ein Gesten-Set. Beachten Sie dabei die für das Spiel wichtigen Kriterien: Das Design jeder Geste sollte möglichst so gewählt sein, dass sie schnell ausführbar ist, motorisch nicht zu schwierig ist, der Benutzer sie sich leicht merken bzw. erlernen kann, sie einen Bezug zu ihrem Effekt hat und sie hinreichend verschieden von den anderen Gesten ist. Inwiefern haben die Vorschläge der Benutzer zu einem geeigneten Gesten-Set geführt und inwiefern mussten Sie sich eigene Gesten ausdenken, um ein gut passendes Gesten-Set zu erhalten? Reichen Sie in einer PDF-Datei die Vorschläge der 3 Test-Benutzer an, sowie Ihren daraus resultierenden eigenen Gestenset ein. Begründen Sie kurz, inwiefern Sie die Vorschläge der Benutzer aufgegriffen haben, bzw. eigene Gesten erstellt haben.

c) Erstellung von Gesten mit dem GestureBuilder. Benennen Sie die Gesten mit den Namen „air", „earth", „fire", „defair", „defearth" und „deffire". Verwenden Sie das in (b) entwickelte Gesten-Set. Die erstellte Gestendatei „gestures" soll mit abgegeben werden. Sie kann mit einem Kommandozeilen-Tool vom Emulator bzw. vom Handy auf den PC kopiert werden: **`adb pull /mnt/sdcard/gestures`**

d) Erstellung der Animationen: (i) für jeden Zauberer eine Animation für die defensiven Zaubersprüche (bleibt als Aura um den Zauberer), (ii) eine Animation für die offensiven Zauberbrüche (wandert vom aussendenden Zauberer zum anderen), (iii) eine Animation des Zauberers selbst, wenn dieser eine Zauberspruch sagt, (iv) eine Animation des Zauberers bei einem Treffer, sowie (v) eine Animation des Zauberers bei 0 Punkten (Spielende). Dies sind für jeden Zauberer 5, also insgesamt 10 zu definierende Animationen. Jede Animation kann mehrere Teilanimationen kombinieren (z.B. Rotation und Translation). Definieren Sie die Animationen komplett innerhalb der xml-Dateien, die in den Ressourcen schon angelegt sind. Die zugehörigen Animations-Klassen sind im Package android.view.animation zu finden. Verwenden Sie die XML-Elemente translate, rotate, scale und alpha. (Hinweis: Die Reihenfolge der XML-Elemente ist relevant. Bei der Drehung eines Objektes um sich selbst, z.B., sollte die rotate- vor einer translate-Animation aufgeführt sein.) Die Aura soll, während sie auf dem Bildschirm ist, pulsieren und sich leicht in der Größe verändern. Ein Angriffs-Spell soll sich während der Bewegung von einem Zauberer zum anderen um sich selbst drehen oder eine andere passende Animation ausführen.

e) Der vom Handy gesteuerte BlackWizard verhält sich momentan noch sehr stumpfsinnig. Er führt einfach alle 4s eine zufällige Aktion aus. Verbessern Sie das Verhalten des BlackWizard, so dass er mit einer gewissen Wahrscheinlichkeit richtig auf Angriffe reagiert. Aber machen Sie ihn nicht so gut, dass der menschliche Spieler (WhiteWizard) keine Chance mehr hat.

f) Neue Benutzer werden zunächst nicht mit dem Gestenvokabular der App vertraut sein. Fügen Sie der App ein Menü hinzu, das einen Eintrag „Show Gestures" hat. Dieser Eintrag soll einen neuen View anzeigen, in dem alle Gesten der Bibliothek mittels einer grafischen Abbildung und deren Namen angezeigt werden, ähnlich wie in GestureBuilder. Um eine grafische Darstellung der Gesten zu erhalten, verwenden Sie die Methode toBitmap() der Klasse Gesture. Sie dürfen zur Bearbeitung dieser Teilaufgabe Teile des Beispielcodes aus GestureBuilder und GestureDemo verwenden, Sie müssen jedoch nicht zwingend ein ListView verwenden.

Abbildung 1. Spielbildschirm (Beispiel). Der offensive Zauberspruch bewegt sich nach links.

Abgabe

Achtung: Plagiate sind verboten und führen zum Ausschluss aus der Veranstaltung!
Das Programm muss kompilieren, sonst wird es nicht korrigiert! Dieses Übungsblatt muss einzeln bearbeitet werden. Exportieren Sie Ihr Projekt aus Eclipse (Export → Archive file) und geben Sie es als zip-Datei zusammen mit dem PDF (Auftabenteil b) und der Gesten-Datei (Aufgabenteil c) bis Montag, den 28.11.2011 um 12:00 Uhr im UniWorX Portal (https://uniworx.ifi.lmu.de/) ab. Sie sollten Ihre Lösung in der Übung vorstellen können.