p5.SOUVI – SOUVI Framework for Processing

Claus Hoffmann

Armenhausgasse 4 1/2
86150 Augsburg
0049 176 219 086 65
claus.hoffmann@
hs-augsburg.de

Moritz Jacobs

Rosenaustr. 50
86150 Augsburg
0049 176 200 418 97
moritz.jacobs@
hs-augsburg.de

ABSTRACT

Nachdem wir während der FMX 2010 in Stuttgart mehrmals das von Daniel Fontes, Björn Gottwald, Veronica Hanglberger, Frank Robnik, Rainer Wagner, Michael Ilg und Peter Alt entwickelte Projekt SOUVI bestaunen durften und uns auch mehrmals mit einigen der Macher unterhalten haben, entschieden wir uns für das Fach i6.inter auch mit diesem technischen Framework zu arbeiten.

Einige der SOUVI-Entwickler wünschten sich, dass wir die Original-Software möglichst nicht modifizieren, sondern lieber etwas eigenes machen; dies kam uns natürlich entgegen.

Die original SOUVI-Software wurde von dem Team in openFrameworks, PureData und unter Zuhilfenahme von OpenCV umgesetzt. OpenFrameworks ist ein sehr vielversprechendes Framework, aber zur Zeit noch relativ unverbreitet. Wesentlich verbreiteter ist Processing, das man auch als Java-Library einsetzen kann, um so in Eclipse o.ä. Arbeiten zu können.

So überlegten wir uns: statt ein kleines Spielzeug umzusetzen, warum schreiben wir nicht die Grundlage für viele weitere Anwendungen die auf SOUVI laufen könnten? So entwickelten wir p5.SOUVI, eine Bibliothek zum Einbinden in Processing. Teil dieses Paketes ist auch ein Tool namens souvi.Calibration. Es dient zum Kalibrieren der Kameramessungen auf die Projektion; die Kalibrierungsdaten werden dabei in XML festgehalten und können dann von der Tracking-Bibliothek gelesen werden.

General Terms

Design, Experimentation

Keywords

infrarot Touchscreen, Processing.org, OpenCV, Java, Blob-Tracking

1. MOTIVATION

Mit dem vorliegenden Framework wollen wir die Basis dafür schaffen, dass nachfolgende Teams interessante und kreative Anwendungen für das Hardware-Touchscreen-System SOUVI (www.souvi.de) auch in der Programimersprache Processing entwickeln können. Die aktuelle Implementierung des Entwicklerteams basiert auf openFrameworks und OpenCV. Processing ist jedoch viel verbreiteter und für Anfänger interessanter.

Um das ganze abzurunden haben wir eine Demo-Applikation geschrieben, die unsere Library verwendet und außerdem, um die Sache etwas aufzupeppen, einen MIDI-Controller als Interface einbindet.

2. Verwandte Arbeiten

Unsere Arbeit bestand hauptsächlich darin, die Funktionaliät des SOUVI in Processing bzw. Java nachzubilden um eine funktionsfähige Library zu schaffen, die später leicht von anderen eingesetzt werden kann. Wir haben weitgehend darauf verzichtet im OpenFrameworks Code unserer Vorgänger zu spicken, um die ganze Sache sportlich zu halten.

Unsere Arbeit unterscheidet sich so inhaltlich kaum von der Basisarbeit die vor uns bereits geleistet wurde, wir haben das ganze nur in einer anderen Sprache implementiert.

3. ANWENDUNGSSZENARIO

Ein mögliches Szenario wäre, dass ein Einsteigerkurs der Programmierung (auch in Java; die Ähnlichkeit zu Processing ist sehr hoch!) eine SOUVI-Anwendung schreibt um schnell sichtbare Ergebnisse zu erzielen, statt monatelang nur auf der Kommandozeile zu programmieren (kann leicht frustrieren).

4. ENTWURF EINES PROTOTYPEN

4.1 Infrarot-Touchscreen

Es wird eine mit Infrarotfilter ausgerüstete Playstation Eye Cam verwendet. Die Rückprojektions-Platte die getrackt wird besitzt an allen vier Seiten eine Infrarotleiste, die zur Mitte hin scheinen. Wird auf die Platte an einer Stelle Druck ausgeübt bricht sich das Licht an dieser Stelle und wird von der Kamera als Punkt wahrgenommen.

4.2 Konzeptueller Aufbau des Prototypen

4.2.1 Playstation Eye

Die Kamera zeigt aufgrund des eingebauten Infrarot-Filters ein sehr leicht trackbares Bild. Wenn man den Raum gut abdunkelt und das SOUVI in eine Verkleidung hüllt, kann man Streulicht vermeiden und die Ergebnisse verbessern.

4.2.2 OpenCV

OpenCV trackt die sogenannten „Blobs“, die in dem von der Kamera gesendeten Bild erkannt werden. Diese sind im Bestfall nur die Fingerberührungspunkte.

4.2.3 TrackAndMap, util.Math

In diesen Java Klassen findet Berechnung für die tatsächliche Projektion statt, also Umwandlung der Trackingdaten in beambare Bilder.

4.2.4 Beamer

Über 2 Spiegel wird die Arbeitsfläche von hinten per Rückprojektion bebeamt.

5. IMPLEMENTIERUNG

5.1 Hard- und Softwareumgebung

Der Code wurde in Eclipse geschrieben. Als Bibliotheken wurden processing, opencv und proMidi (für die Demo-Applikation) verwendet.

5.2 Kern der Implementierung

Das Herzstück des Frameworks ist die Zusammenarbeit zwischen den Klassen Calibrate und TrackAndMap. In der Klasse Calibrate werden zwei Trapeze bestimmt (Bild der Kamera und Projektionsfläche) und in einer XML-Datei gespeichert. Somit muss diese Klasse nur einmal nach dem Aufbau aufgerufen werden. Aus diesen XML-Daten berechnet dann die Klasse TrackAndMap die Position der Projektion, indem sie das Kameratrapez auf das Projektionstrapez umrechnet.

5.3 Umfang des Prototypen

Der Prototyp beherrscht bereits das Tracken eines Fingers auf der Projektionsfläche. Beachten Sie bitte auch das beigefügte Demo-Video. P5.SOUVI ist eine lauffähige Processing-Library, die ohne weiteres in Processing-Sketches importiert werden kann.

6. EVALUATION

6.1 Ziele der Evaluation und Methodik

Es wäre wichtig herauszufinden, welche Probleme sich in der Verwendung unserer Bibliothek und unseres Kalibrierungstools auftun und wie man diese vermeiden könnte.

6.2 Durchführung der Evaluation

Um dies zu gewährleisten sollte über die nächsten Jahre beobachtet werden, wie andere damit zurecht kommen. Eine schnelle Nutzererhebung ist hier nicht sinnvoll.

6.3 Interpretation der Evaluationsergebnisse

Durch mangelnde Dokumentation der einzelnen Komponenten ist es so, dass unsere Arbeit von „Trial and Error“ gekennzeichnet war. Dies und der enge Zeitrahmen sorgte dafür, dass wir die Bibliothek weniger modular und allgemein halten konnten, als wir wollten. Dies wird in Zukunft Probleme aufwerfen, die wir zukünftig noch behandeln sollten.

7. ZUSAMMENFASSUNG

7.1 Erzielte Ergebnisse

Es war ein befriedigender Moment, als der weiße Punkt auf dem Screen das erste mal dem Finger des Testers folgte. Abschließend lassen sich nur einige wenige Erkenntnisse festhalten:

- Die JVM ist ein massiver Flaschenhals bei Anwendungen, die komplexe Bildberechnungen anstellen müssen.
- OpenCV ist für Anfänger nicht trivial zu installieren
- Processing ist ein gelungenes Toolkit, aber immer noch weit von perfekt entfernt, da an vielen Stellen, gerade für Anfänger, immer noch zu wenig sinnvolle Abstraktion von der Muttersprache, Java, zu merken ist.

7.2 Erweiterungsmöglichkeiten / Ausblick

Zukünftige Verbesserungen/Erweiterungen könnten sein:

- Multitouch
- bessere Intgration mit der Processing-PImage-Library

8. DANKSAGUNG

Großer Dank geht an alle Leute die ursprünglich an dem souvi Projekt beteiligt waren, dass sie uns die Möglichkeit boten an ihrem Projekt weiterzuarbeiten, sowie für ihre Unterstützung:

Daniel Fontes

Björn Gottwald

Veronica Hanglberger

Frank Robnik

Rainer Wagner

Michael Ilg

Peter Alt

Betreuer: Prof. Thomas Rist

9. REFERENZEN

- OpenCV for Processing

  http://ubaa.net/shared/processing/opencv/
- Processing Reference

  http://www.processing.org/reference/
- Java Documentation

  http://www.oracle.com/technology/tech/java/index.html