Praktikum #4 & #5

Multi-Touch und Multi-User Interfaces

Björn Frömmer

Ziele von Praktikum #4 & #5

- Implementierung einer eigenen Multi-Touch Anwendung, welche die Daten von dem in Praktikum #1 - #3 entwickelten Tracker bekommt
- Interaktion mit grafischem Interface (keine Konsolenanwendung, siehe “Natural User Interfaces”)
- Gestenerkennung (kontinuierliche und diskrete Gesten)

Leitfaden Praktikum #4 & #5

- Ziele des vierten Praktikums:
  - Implementierung einer einfachen Multitouch Client-Anwendung unter Verwendung der TuioClient API
    - Sie können als Einstieg den Sourcecode der TuioDemo verwenden, oder etwas eigenes erstellen, hier gibt es keine Vorgaben
  - Integration des TUIO Frameworks in den Client
  - Auswerten der TUIO Events (add, update, remove)
  - Testen des Clients mit dem eigenen Tracker (Praktikum 3), und/oder einem Touch Simulator (http://tuio.org/?software)
- Ziele des fünften Praktikums:
  - Fertigstellung des eigenen Multi-Touch Clients
  - Integration von Gestenerkennung in den Client (kontinuierliche und diskrete Gesten, siehe hierzu auch “Bewertung des Praktikums.pdf”)

Code Example:
Virtual TuioClient Memberfunctions

```
class myAppClass : public TuioListener
{
        public:

        void addTuioCursor(TuioCursor *tcur);
        void updateTuioCursor(TuioCursor *tcur);
        void removeTuioCursor(TuioCursor *tcur);

        [...]
}
```

Code Example:
Implement the virtual Memberfunctions

```
void myAppClass::addTuioCursor(TuioCursor *tcur) {...}

void myAppClass::removeTuioCursor(TuioCursor *tcur) {...}

void myAppClass::updateTuioCursor(TuioCursor *tcur)
{
        […] = tcur->getCursorID();
        […] = tcur->getSessionID();
        […] = tcur->getX();
        […] = tcur->getY();

        […]
}
```

Code Example: TuioClient Implementation

```
int main(int argc, char* argv[])
{
        […]

        myAppClass myClient;
        TuioClient client(port);
        client.addTuioListener(&myClient);
        client.connect(true);
}
```