BP WS 2015/2016 – Team20
Dokumentation für die Auftraggeber

Was macht das Programm?

- Das Programm tracked bis zu drei Personen gleichzeitig und speichert diese mit einer ID im System ab. Von den Personen werden mit Hilfe der ID User-Einträge erstellt, die unter anderem das Skelett enthalten, welches von der Microsoft Kinect geliefert wird. Das Programm erkennt die Farbe der Nutzer mit Hilfe der writableBitmapEX-Bibliothek, wodurch Personen vom System als aktive Nutzer mit ihren bereits bekannten Daten wieder erkannt werden. Das Programm erkennt bei einer Touch-Eingabe von welchem Nutzer diese stammt, wodurch Daten individuell gespeichert und angezeigt werden können.

- Weiterhin kann das Programm auch mit zwei Kameras genutzt werden, in dem eine Kamera an einen anderen Rechner angeschlossen wird. Durch den tcpHandler kommunizieren die zwei Rechner(ein Client und ein Host) und der Client versendet seine gesehenen Daten (primär User-Daten und ihre relativen Handpositionen zum Bildschirm) über den tcpHandler zum Host. Die Daten werden im System des Host dann entsprechend verbunden, sodass das System die bestmöglichen Daten über die User empfängt.

- Dazu gibt es die Demo1, welche die gesammelten Daten nutzt und den Nutzern beim Zeichnen eine Farbe zuordnet. Die Daten werden solange angezeigt, solange der Nutzer im Kamerablickfeld ist. Sollte der Nutzer wieder ins Kamerablickfeld zurückkehren, werden seine Zeichnungen bei Erkennung wieder angezeigt.

Grundlegende Funktionalitäten und wo sie sich im Programm befinden:

- Das Event für die Erkennung der Bodies nennt sich Reader_OnBodyFrameArrived(Klasse MainWindow). Hier wird bei jedem neuem Frame überprüft, ob sich die Anzahl der Nutzer verändert hat.(durch countActiveBodies() der Klasse MainWindow) Wenn dies der Fall ist, wird usertracking() der Klasse MainWindow aufgerufen und es wird überprüft, ob Nutzer neu vom System erkannt wurden oder sich bereits im System befinden.

- Falls eine zweite Kinect benutzt wird, werden nach dem usertracking() die empfangenen Daten durch die Funktion mappingHandling() der Klasse mappingHandler mit den Daten der eigenen Kinect zusammen geführt. Hierzu wird überprüft, ob Nutzer von Host und Client mit gleichen Farben erkannt wurden.

- Zur Farberkennung dient das Event Reader_ColorFrameArrived(Klasse MainWindow), welches die ColorFrames der Kamera stetig aktualisiert. Um dem User eine Farbe zuzuweisen, wird die Funktion colorHandling() der Klasse colorHandler verwendet. Bei colorHandling() wird die Farbe in einem Bereich in der Mitte des Körpers(Spinemid) Pixel für Pixel untersucht und die häufigst auftretende Farbe dem Nutzer zugeordnet. Für den Vergleich wird die RGB-Farbe in den HSL-Farbraum umgewandelt, um Lichtunterschiede besser behandeln zu können.
- Das Event für die Touch-Inputs wird Touch_FrameReported(Klasse MainWindow) genannt. In diesem Event werden alle TouchPoints, die auf dem zentralen Canvas des Hauptfenster erkannt werden, behandelt. Jedem TouchPoint wird ein Nutzer zugeordnet, in dem die Kamera-Daten von den Händen der Nutzer in Relation zum Bildschirm in Touchscreen-Koordinaten umgewandelt werden und somit dem nähesten Nutzer der TouchPoint zugeordnet werden kann. Dies wird durch die Funktionen

convertCoordinatesKinectToScreen() und compareCoordinatesKinectToScreen der Klasse mappingHandler realisiert, welche durch die Funktion findUserOfTouchPoint() der Klasse MainWindow aufgerufen werden.

- Beim Senden der Daten über den TcpHandler werden die Daten des Clients in einen String umgewandelt, welcher einfacher zu schicken ist. Dieser wird vom Host in die richtigen Daten dekodiert und als Felder vom MainWindow gespeichert.

- Die Demo1 ist ein einfaches Malprogramm, welche die TouchPoints, die mit den User-Daten verbunden wurden, nutzt, damit diese je nach User unterschiedlich angezeigt werden können. Dabei wird die Funktion TouchFrameDrawingDemo() dazu genutzt, die Linien auf dem Canvas zu zeichnen und diese in Arrays im Gedächtnis zu behalten. Falls ein User aus dem Blickfeld der Kamera tritt, können so seine Daten vom Canvas entfernt werden, damit für den nächsten User mehr Platz ist. Die Funktion updatecanvas() ist dafür zuständig. Weiterhin existiert die Funktion updatecanvas2(), welche neue Nutzer und wiedererkannte Nutzer speichert. Bei wiedererkannten Nutzern werden seine vorherigen Zeichnungen wieder auf dem Canvas angezeigt. Falls ein Nutzer seine Zeichnungen löschen möchte, kann er den Reset-Button der Funktion deletebutton() nutzen. Dieser löscht nur die eigenen Daten des Nutzers.

- Am Anfang des Programms wird das StartWindow aufgerufen. Hier werden vor dem Aufruf des Hauptfensters wichtige Daten eingestellt. So muss der Benutzer bei Nutzung von zwei Kinects die IP-Adresse und den dazugehörigen Port des Hosts angegeben. Weiterhin muss der Benutzer einstellen, ob der Rechner der Client oder Server sein soll. Wird nur eine Kinect genutzt, muss der Nutzer als Client angemeldet sein. Weiterhin muss der Nutzer das SettingsWindow aufrufen, um die Position der linken unteren Ecke und der rechten oberen Ecke von der Kinect des Rechners aus einzugeben. Diese müssen dabei reelle Zahlen sein. Die Koordinaten der Ecken findet man am besten durch einen Console.Write()-Aufruf im Touch-Event, indem man sich die CameraSpacePoints der linken Hand beim Berühren der Ecken (innerhalb des Canvas) ausgeben lässt.

Wie man eine eigene Demos aufruft:

- Eine eigene Demo sollte als separate c#-Datei gespeichert und am Anfang vom MainWindow initialisiert werden.
- An den Stellen, an denen die Demo1 aufgerufen wird, sollte die eigene Demo aufgerufen und die Referenz der Demo1 auskommentiert werden. Diese Stellen befinden sich in Touch_FrameReported und Reader_OnBodyFramedArrived.
- Weiterhin muss das Array mergedTouchPoints für das Nutzen der TouchPoints, die mit den User-Daten verbunden wurden, der Demo übergeben werden.
- Alternativ kann die eigene Demo extern aufgerufen werden, in dem das MainWindow ein durchsichtiges Canvas besitzt und das StartWindow neben dem MainWindow die Demo aufruft. Dabei sollte das neue MainWindow der Demo übergeben werden, damit diese alle neuen Änderungen von mergedTouchPoints nutzen kann.