Finger tracking using touchless input devices

# Initial Goal

Goal of this project is the evaluation of finger tracking capabilities of touchless input devices. Primarily the Leap Motion will be investigated. Research of additional devices will also be conducted (e.g. Kinect) and compared with the Leap Motion.

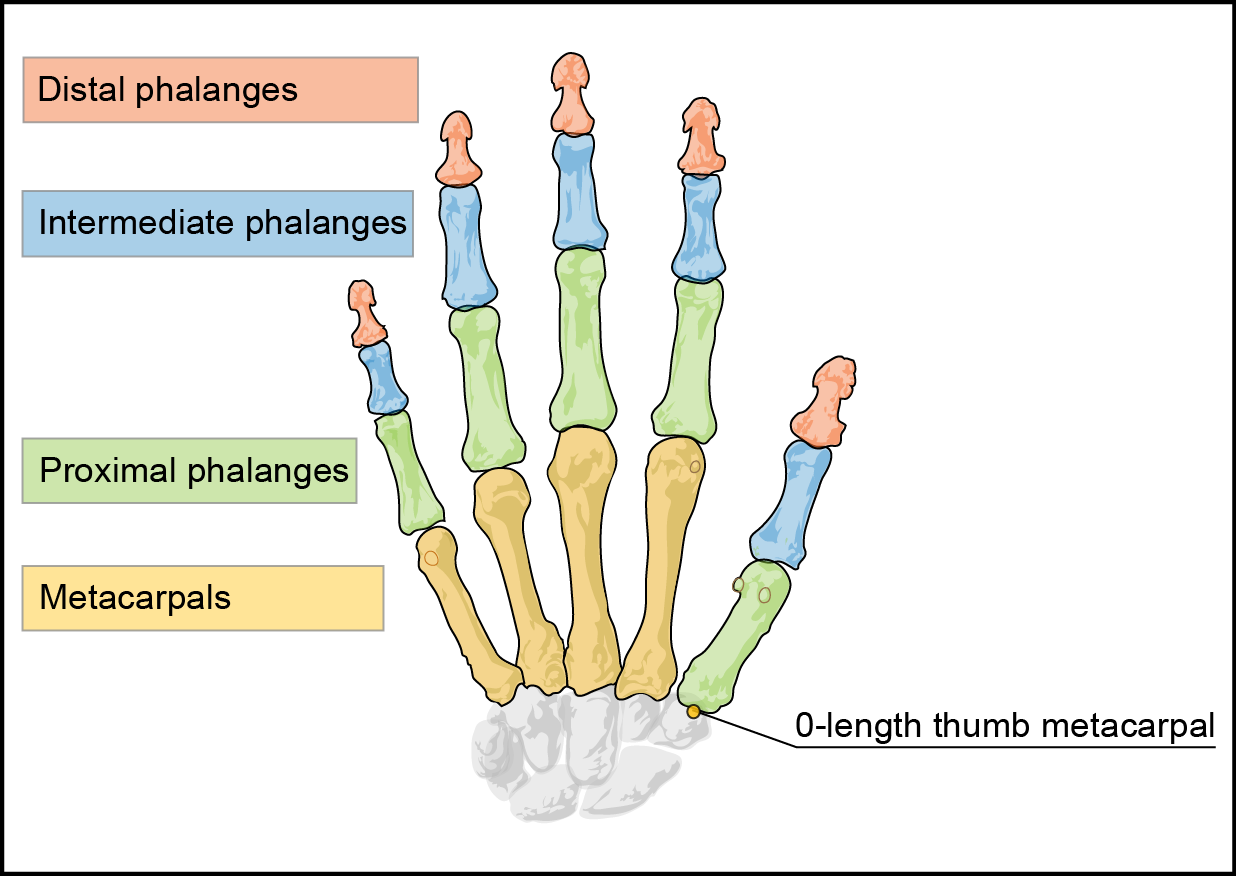
# Planned Milestones

1. First, a short survey about devices suitable for finger tacking is created.
2. Based on these devices, a small framework will be developed for generating the bounding volume hierarchy of the user's hand. This milestone includes some basic visualization for the bounding volumes.
3. Thirdly, the bounding volumes from the hand are loaded into a physics simulation environment. This includes a simple scenario where the user has to grab and move an object (bowls example).
4. After the simple scenario, more advanced scenarios with complex interactions and sophisticated physics constraints will be created (e.g. solving a Rubik’s Cube).
5. Based on the data gathered in these scenarios, a conclusion about the capability and suitability for finger tracking with the Leap Motion and other touchless input devices is drawn.

# Survey of devices capable of finger tracking

## Leap Motion

Skeletal tracking of fingers is officially supported by Leap Motion SDK in version 2.0. The Leap API provides information about each bone of each finger, most notably the knuckle positions and the orthonormal basis of the bone's center (a matrix that describes the bones transformation relative to the origin).



## Microsoft Kinect

Finger tracking is not supported by Microsoft's Kinect SDK, but access to depth image stream allows using third party libraries, for example (suggested by Prof. Kurschl):

* frantracerkinectft
* KinectLibrary
* CandescentNUI

## Conclusion

Due to the superior finger tracking capabilities of the Leap Motion, Kinect support has been dropped in favor of a more in-depth investigation of the new Leap Motion 2.0 SDK.

# Demo Application Infrastructure

The demo application provides the user with a platform to interact with a virtual world with a touchless user device. The user can load from a set of different scenes which each contains of multiple *scene objects*. To interact with the scene, the user has to use a touchless input device (e.g. the Leap Motion) to track the motion of his hands, which are then placed into the scene.

## Architecture Overview

The application infrastructure is designed to offer extensibility for multiple devices. Therefore it is organized in multiple layer to abstract different kind of data. The Core layer contains device independent data structures for representing the status of a user’s hand.

The users hand is then converted into bounding volumes (Physics Hand), which are placed into the Physics World (i.e. the scene). The scene, including the physics-model of the hands, is simulated with the Bullet Physics Framework. Bullet itself sets the *motion state* of all objects in the physics world, which describes the position and rotation of an object in the physics world.

Later, all objects in the physics world as well as the hand are rendered depending on their motion state (Hand Renderer, World Renderer). For the visualization, the physics shapes are converted into triangles and then rendered via DirectX.

## Data Flow

The application consists of 4 major stages. The first stage contains the raw device data from the touchless device (i.e. the Leap Motion). This raw data is processed and converted into device-independent data, which contains data structures for hands, fingers and bones. For the next stage, bounding volumes for each bone and for the palm are generated. The bounding volumes from the hand as well as other *scene objects* are added to the physics simulation. With the Bullet physics library, the physics world is simulated and the *motion state* of each object is calculated. The visualization stage draws the 3D scene based on the results of the physics simulation.

# Physical accurate hand reconstruction using Bullet

Bernhard, from Leap data to the rigid bodies

# Test scenarios

Screenshot of each scenario with a short description about what shall be especially tested with this scene.

# Problems and conclusion

Bla bla bla,

We thought this shall be easy

Bullet is extremely badly documented.

Even simple scenes turned out to be extremely challenging,

Problems with friction and movement of kinematic objects (the hand)

LeapSDK+