# Studienarbeit – I believe I can fly V2.0

Henri Kohlberg, Oktober 2018 – Mai 2019

### KW 41:

* Start der Studienarbeit
* Testen der bereits bestehenden Lösung (V1.0)
  + Funktioniert einwandfrei nach Installation des [Microsoft Kinect 1.8 SDKs](https://www.microsoft.com/en-us/download/details.aspx?id=40278)
  + Vorwärts und rückwärts Fliegen ist nicht sehr genau, seitliche Bewegungen ohne Probleme
  + Verwendung verschiedener VR-Brillen über HDMI ohne Probleme (auch mit eigener Fatshark HD3)
* Upload von Code in [Git-Repository](https://gitlab.com/Burdel/ibicf)

### KW 42:

* Versuch die zweite Studienarbeit (V1.5) aufzusetzen
  + Aufsetzen verschiedener Systeme von verschiedenen Medien (USB-Sticks, Backups, ...)
  + Kein Auffinden der Ubuntu 14.04 Version, welche den Aufzeichnungen nach alle ROS Packages enthalten soll
  + Auffinden einer defekten Ubuntu-VM auf ThinkPad Laptop (Boot Medium fehlt)

### KW 43:

* Wiederherstellung des bisherigen Ubuntu 16.04 Images auf MSI Laptop
  + Installiertes Windows wurde entfernt, da keine Daten gespeichert waren und Laptop nicht in Verwendung war
  + Ubuntu Image scheint bis auf AirSim-Demo-Projekt leer zu sein
* Setup von virtueller Maschine mit Ubuntu 18.04, ROS, Unity und AirSim
* Anmeldung Studienarbeit (Moodle)
* Einarbeitung in ROS (Offizielles ROS Tutorial)

### KW 44:

* Ausprobieren von ROS Version auf Ubuntu 14.04
  + Startup Skript automatisiert
  + Funktioniert ohne Probleme
* AirSim hat nun [ROS Integration](https://github.com/Microsoft/AirSim/blob/master/docs/ros.md) (ROS Kinetic, Ubuntu 16.04)
  + Integration in AirSim sollte möglich sein

### KW 45:

* Windows 10 auf MSI Laptop installiert
* Ubuntu 16.04 auf MSI Laptop installiert, Dualboot (inkl. AirSim, UnrealEngine, ROS Kinetic)
* Erste Version von Skeleton-Tracking implementiert
  + OpenCV Library mit pre-trained COCO Netz
  + Problem: zu langsam, da nur mit CPU gerechnet wird
  + Keine sinnvolle Verwendung möglich

### KW 47:

* OpenPose installiert
  + Support für GPU Berechnungen
  + Python API
* Erste Implementierung von OpenPose vorgenommen
  + Echtzeit-Erkennung durchaus möglich (~15 FPS)
  + Erkennung sehr zuverlässig, auch bei schlechten Licht-Verhältnissen

### KW 48:

* Aufbau der ROS Nodes überlegt
* Einbettung von Personen-Erkennung in ROS Node
* Koordinaten werden in ROS Topic „/person\_detection“ veröffentlicht und können weiterverarbeitet werden
* Schreiben des monatlichen Berichts für November 2018

### KW 49:

* Ausarbeitung einer Steuerungs-Logik für 2D-Koordinaten
* Debugging-Option hinzugefügt
  + Speichert Bilder und Keypoints extern
  + Vorbereitung für Gestenerkennung

### KW 01:

* Exportierte Daten (JSON) einlesen
* Implementierung einer ersten Gesten-Erkennung
* Schreiben des monatlichen Berichts für Dezember 2018

### KW 05:

* Anfang von Studienarbeit
* Schreiben des monatlichen Berichts für Januar 2019

### KW 08:

* Integration der Positionserkennung in ROS
* Fine-Tuning für Positionserkennung
  + Zuverlässige Erkennung aller Posen mit gewissen Spielräumen
* Kurzschließen mit Masterand, welcher auch mit OpenPose arbeitet
* Reinstallation von Unreal Engine & AirSim da Projekte nicht geladen werden konnten
  + Erfolgreich nach erneutem Kompilieren & Bauen
* Implementierung als Proof of Concept in AirSim
  + Takeoff und Landung funktionieren, Pitch-Achse nur bedingt
  + Genaue Steuerung muss noch implementiert werden
* Aktueller Stand (Gespräch mit Herr Strand)

### KW09:

* Volle Integration in AirSim (Forward, Turn Left/Right, Up, Down, Liftoff, Hold)
* Verwendung der Welt „Neighborhood“ und „Factory“
  + Realistisch Welt
  + Leichte Leistungseinbußen, welche bei Verwendung der FPV Brille unerfahrener Benutzer zu Nebenwirkungen führen könnte
* Schreiben des monatlichen Berichts für Februar 2019

### KW10:

* Einlesen zu Fuzzy Controllern
* Entwicklung eines Fuzzy Controllers für die Gesten Erkennung
* Implementierung des Fuzzy Controllers in die bestehende ROS Umgebung
  + ROS Node pose\_detection kann jetzt nicht nur noch ein Steuersignal ausgeben, sondern mehrere Signale mit unterschiedlichen Stärken
  + Signale müssen in AirSim-Connector ordentlich verarbeitet werden und in ein Signal für AirSim umgewandelt werden
* Anfang der AirSim Steuerung durch den Fuzzy Controller

### KW12:

* Fertigstellung der Fuzzy Steuerung und Implementierung in AirSim
* Ausgaben an den Benutzer um ihn durch die Kalibrierung zu leiten

### KW13:

* Anpassung der Start und Stop Prozedur
* Schreiben der Schnellstartanleitung
* Mehrere Personen im Bild => Mittigste Person nehmen (Abstand Bildmitte zu mittlerer Hüfte)
* Kamera-Bild mit Skelett während Simulation zeigen (Windowed Mode)
* Feintuning der Steuerung
* Shortcut für Simulationsreset eingefügt (R -> Reset Kalibrierung und Position)
* Umbenennen der ROS Nodes in einheitlichen Namespace

### Planung:

* Anbindung der 3D-Kamera Intel RealSense
* Anbindung der echten Drohne
* Umbenennung der ROS Channels zu Namespace