Autonomous Flying Mini Drone-ball in a Multiplayer AR Superhuman Sports Game

Yixuan Liu* Min Ting Luong* Christian Eichhorn† David A. Plecher‡ Gudrun Klinker§

Technical University of Munich

Abstract

Superhuman sports are somatic activities combined with technologies like mixed reality, extending players' ability at the playground by using wearable gears. A superhuman sports ball game is designed for players able to interact with a tangible drone ball, where its 6D pose can be tracked using onboard LEDs.

This paper explains how we developed a drone, which autonomously flies and safely interacts with players and constructions in the game, using techniques for obstacle avoidance, target tracking and decision making. We have overcome many constraints. The drone is required to carry many sensors, extra computing boards, to fly long enough to be integrated into the game and additionally, all parts have to fit in a small cage. Meanwhile, relatively lightweight and cheap companion computers with miniature sensors are not always powerful enough to run the existing open-source detecting and tracking algorithms. Also, since the drone will be caught by hand, two layers of cages protect the players' fingers, and a force-resistant touch sensor detects the catch. Furthermore, the in-game interaction is set up, including a tripod stand with a flexible goal ring and goal detection sensors.

Index Terms: Augmented Reality, Tangible AR, Autonomous Flying, UAV, Drone, Catchable Drone Ball, Superhuman Sports Games, Interaction Games Concept——

—

1 Introduction

This paper addresses the potential of a mini autonomously flying drone, which helps an Augmented Reality (AR) game be more tangible and immersive for players in-game. This game belongs to the genre Superhuman Sports Game (SSG), where human players overcome natural limitations and obtain extended abilities with new technologies, such as Virtual Reality (VR) and AR. At the same time, to still have a close-to-reality gaming experience, somatic and haptic feedback are also crucial.

The game has two teams and the objective of each team is to catch the drone and throw it in such a way, that it flies through one of the goals at the end of the playground. Our project aims to build such a drone, that can recognize and fly away from obstacles and towards human players, as well as goals that can detect the drone flying through.

The drone must not only be smart but also robust enough to withstand any throw, however still safe so that it does not hurt anyone. After three major version updates, we are proud to present the Golden Snitch in Figure 1. It can hover on the spot without GPS and can detect human pose and walls, then move accordingly based on defined game rules. It is feasible for players' AR headsets to track the drone by detecting its LED pattern. The game goal rings can also detect the drone, when it flies through.

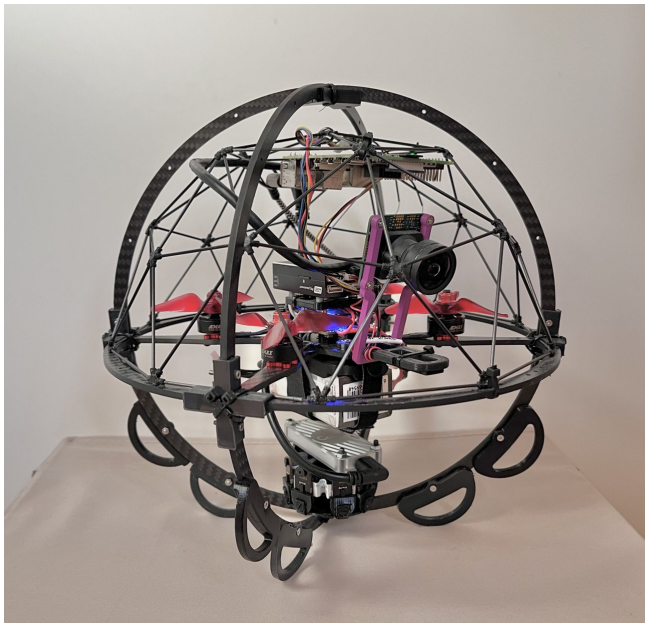


Figure 1: Indoor autonomously flying quadcopter Golden Snitch ensures Superhuman Sports Game a safe and tangible AR experience.

This paper includes the following main contributions:

- Explaining the composition of our drone, that matches the requirement of the desired indoor SSG.
- Describing the onboard sensors and software for autonomous flying.
- Exploring the in-game interaction supports, how our drone can be tracked and how players get points and win the game.

2 Related Work

2.1 Superhuman Sports Games

SSGs are somatic activities combined with technologies like AR and VR, extending players' ability at the playground by using wearable gears. Artists, athletes, engineers, scientists and more people with various skills and different backgrounds gather together and engage to create new-era sports.

One famous organization in this area is the *international Superhuman Sports Society* (S3) [48] originated from Japan. S3 was founded in 2015 and is now flourishing. Like other Japanese cultural influences, S3 is also a collection of entertainment, technology and pop culture [44]. Students and industries are also actively involved. For instance, TU Delft hosted international challenges for SSG concepts [23].

*e-mail: yixuan.liu@tum.de, minting.luong@tum.de

†e-mail: christian.eichhorn@tum.de

‡e-mail: plecher@in.tum.de

§e-mail: klinker@in.tum.de

2.2 Augmented Reality in Superhuman Sports Games

Although there is not a large quantity of SSGs in AR yet, its developer society and market are emerging rapidly.

The S3 has gathered more than a dozen of developer teams to create SSGs. Some of them are involved with AR elements. One great example is "Hado" [46], a 3-on-3 battle, in which players equipped with a head-mounted display (HMD) and arm sensors throw energy balls and set barriers to reduce opponents' life.

Another AR SSG of the S3, "Hado Cart" [47], seems to be inspired by Mario Cart. In this game, players sit on scooters and drive around to hit other players and to collect coins.

At TU Delft, an adventure shooter game called "Superhuman Training in Augmented Reality" (STAR) [33] is implemented on the Microsoft Hololens. In their AR headsets, players have to work together as a team to destroy an energy core at the destiny. On the way, players should avoid enemies and collect energy points while passing through a narrow path above the lava. The team interacts in multiplayer mode online.

2.3 Existing Drone Solutions in Sports Games

First person view (FPV) drone racing games are becoming more popular. At the same time, it has a great potential to be a future E-sport, where players, who are wearing AR or VR headsets compete with drones. Take the app *AR.Rescue 2* as an example, which is an AR drone game [11]. The pilot has to fly through an area, collect things and destroy appearing monsters that are augmented on a display on the way.

Furthermore, the development of a protecting cage around drones made drone racing even more competitive, as one can not only assemble and configure their own drones, but also design their own effective cages. For example, the *Graupner Sweeper Droneball* has a crash resistant cage around the drone to protect itself from any harm after a crash landing [28]. Using a protective cage, drones can engage in sports, where they come into full contact with each other, such as *Drone Soccer*. In *Drone Soccer*, each team tries to fly through the other team's gate. A goal can be prevented by a drone flying against another drone to push the latter one away from the gate [55]. A different interesting example is the *Flynova Pro*, which is not a drone in fact, it is a flying ball. With the *Flynova Pro* you can throw the ball with similar super moves like in our tangible AR superhuman sports game "Catching the Drone" [24], so to speak, if you tilt the ball up and left, the ball will fly a left turn [19].

Nowadays, drones are used not only in sports but also in numerous other fields. The size of a drone varies from big (drone taxi) to very small (mini drones). They can be little or big helpers and, with the advancement of technology, the possibilities seem to be only limited by one's imaginations. In cinematography they are used to create magnificent cinematic videos of the environment from impressive angles of view. Tobias Nägeli et al. propose an method for planning the flight path of multiple filming drones in real-time, taking the unpredictable movements of the actors in consideration [40]. Military drones are used to collect information [9]. Drones could save lives after a natural catastrophe by consuming less time to search for human bodies. The *DroneAID* system searches for human bodies and informs the rescue teams about their whereabouts with a live stream, so they can help the victims as fast as possible [50]. Drones could also be utilized to deliver medical supplies to hardly reachable environments [45] [49] or to observe wildlife unobtrusively in their natural habitats [32] [43]. On the other hand, social drones are developed for places that are inhabited by people [17], like the drone that films sports participants during their run such as on the annual TUM Campuslauf. In the paper of Tominaga et. al "Around Me: A System with an Escort Robot Providing a Sports Player's Self-Images" [56] a robots that is always ahead of the sport player records and analyzes their form. The robots will display an overlay of the correct form or an instructor can remotely draw his comments onto its display. In a similar way, a drone with a display and camera could act like a coach or a fitness companion and motivator such as the in "Jogging with a Quadcopter" from Mueller et. al [39]. There, a quadcopter accompanies joggers by flying in front of them at a certain pace and functions as a "pace keeper". The quadcopter will guide the joggers through a pre-planned jogging path. Likewise, a drone could act as a direct guide for pedestrians [20].

There are sports-related preexisting drone solutions with tangible human-drone interaction with in close-range or direct interaction with the drone, some like in the game "Catching the Drone" from Eichhorn et. al [24]:

In the paper "Boxing against drones: Drones in sports education", an autonomously flying drone is an interactive training partner in boxing. Considering the fact that a drone is breakable, this flying training partner is suitable for low impact training like shadow boxing, condition training, strategic training, walking and defense training. The drone determines the boxer's location and the impact through sensors that are integrated into the boxing gloves. The training starts, when the coach presses the start button of a training program on the app. After that, the drone executes prearranged movements. [57]

With "Drone Chi" users can practise Tai Chi with a small, dainty mini drone in a shape of a lotus flower. The drone is controlled with two curved small hand pads, one on each palm. An invisible line is drawn between the pads, on which the middle point will be calculated. The drone moves accordingly to its offset distance to the middle point of the line between the palms. The drone has two modes. In the lead mode, the drone is following the hands, if they are within a 20 cm offset. If the drone notices that it is too far away from the user or it moved too fast, it will slow down and slowly flows away for the user to re-collect. In the follow mode, the user follows the drone with their hands and the drone will gradually fly bigger circles. [36]

The paper "Tactile Drones - Providing Immersive Tactile Feedback in Virtual Reality through Quadcopters" proposes to use small drones for tactile feedback while a user is visually and auditory engaged in VR. In *TacticleDrone*, a VR player goes through a jungle, he will be attacked by objects in 3 different scenarios. In the first, he is attacked by a bumblebee, which is simulated by one drone flying against the player. In the second scenario, creatures are aiming at the player with arrows, which is achieved by having a drone equipped with a tip poking the VR user. Lastly, there are bricks, wood and skulls swirling around the scene. So an accelerated drone will hit the player with a larger force than in the other two scenarios [35].

3 Mini Drone-ball's Construction

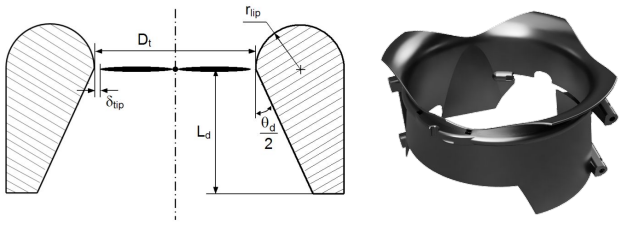
To support the tangible AR concept in [24], a drone-ball in appropriate dimensions, a.k.a. with a size similar to a handball, has been built. As this indoor-flying drone shall be safe when being close to players, even ready to be caught in hands, a lightweight protection cage is needed. It needs to fly as efficiently as possible so that its flight time lasts the longest. In this section, we describe how we reduce the weight and combine all onboard parts into a tiny drone-ball cage.

3.1 Mechanics Design

In the original concept, Golden Snitch Version One (GSv1), we have designed and mounted 3D-printed ducts around the propellers. On the one hand, the ducts are physical protection for users who will catch the drone by hands; on the other hand, ducts increase the lift and stabilize during hovering.

To decrease the total onboard weight as much as possible, we kept the ducts as thin as feasible so that our 3D-printer Prusa is still

able to slice and print the wall in two layers, almost at the minimum limit for the desired wall strength.

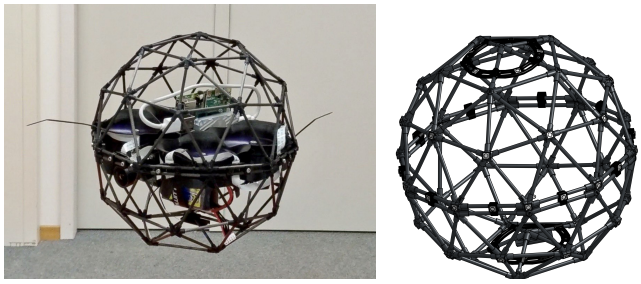


(a) Propeller duct design theory [41] (b) Propeller duct design of GSv1

Figure 2: Left: The theoretical perfect duct for hovering in a drawing, which is taken as a reference for our propeller duct design. Right: On the 3D-printed ducts we made many compromises. They are not only used to increase lift and to protect fingers from spinning propellers while catching the drone, but they are also connecting the main frame of GSv1 and its cage, as well as supporting the main camera.

For the first cage design of GSv1 we took Flyability's indoor drone Elios 2 [25] as a reference, which has a form similar to the C-60 molecule. Phoenix Robotics member Dominik Weiss designed several types of connectors with different angles, then milled hollow carbon fiber rods and glued them into the 3D-printed connectors. To reduce the weight as much as possible and to increase the flight time, Dominik has designed the connectors as thin as possible. However, during flight tests, several crashes revealed that the connectors of polylactic acid (PLA) were not crash-resistant. After every crash new connectors were required. Re-printing and re-mounting led to time lost since this blocked further flight tests. Also, a chain of distance sensors was attached to the middle rings of the cage. The sensor holders were also holding the two cage halves. After a crash, one distance sensor cable was dragged into the propellers and was therefore damaged. The latest version of sensor tower is mentioned in subsection 4.1.

The Golden Snitch Version Two (GSv2) uses connectors that were printed using thermoplastic polyurethane (TPU), which is more elastic and barely breaks even while being pulled with an extremely large force. Also, the ducts out of PLA were removed, as they frequently broke.

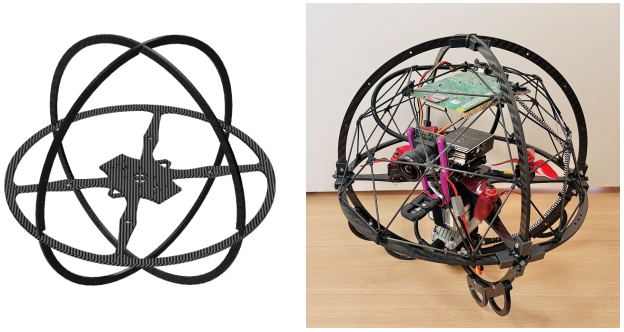


(a) GSv1 in flight test (b) GSv2 cage with embedded LEDs

Figure 3: Left: The flying GSv1 in the flight test room at the Institute of Flight System Dynamics, TU Munich. Right: The cage v2 has LEDs embedded in the 3D printed connectors rendered in Autodesk Fusion360.

The current version, Golden Snitch Version Three (GSv3) has an outer skeleton added to the second version. It consists of three rings cut from 2 mm-thick carbon fiber sheets, one of the rings is just the original main frame being extended at the end of arms. Intended to have the level of propellers in the middle layer, where the motors sit, the frame and the horizontal ring are 5 cm lower than the mid-layer. Therefore, the lower cage and the ducts were removed. It is now much more convenient to install the drone and to change the batteries. The upper cage of GSv2 is kept as the inner cage, which will still protect the players' fingers.

To keep the component with the most weight, which would represent the gravity center, as close to the frame center as possible, the flight controller (FC) is moved up to the motors' center. The companion computer (CC) is now zip-tied onto the inner cage. Unexpectedly, its big flat green surface increased the chance to be detected at the goal rings, since the sparse dark cage can barely be detected by IR or ultrasonic sensors as shown in subsubsection 5.4.3.



(a) Outer skeleton of GSv3 (b) GSv3

Figure 4: Left: The main frame v3 is rendered in Autodesk Fusion360. The main frame v1 is extended with one outer ring. With another two vertical rings together, the frame also becomes an outer skeleton to protect the inner cage, which is shown in Figure 4b. The outer rings are connected using 3D printed connectors out of TPU and metal screws. Right: The current construction of GSv3 with Raspberry Pi 4B as its CC on the top. The computer is fixed with zip-ties to reduce support weight.

3.2 Electronics Layout

Figure 5 shows the main electronics onboard. The following list includes a description of the main components except for external sensors of the named computers. The sensors for autonomous flying are described in subsection 4.1, and the sensors for in-game interactions are mentioned in section 5.

- **Flight controller:** The most crucial and essential component for an autonomous flying drone is the FC. We chose Holybro PixHawk 4 Mini [29] for the flight control framework ArduPilot [14], which is an open-source autopilot system. The FC collects data from all accessible sensors, and estimates its state, for example, its current height, ground speed and distance to a target or an obstacle. Meanwhile, its information influences its control by sending low-level signals to the speed controller to control the motors. Besides, it also allows communicating with remote controllers, and also ground stations.

- **Companion computer:** Additional to the FC, we need another computer to process visual data. Compared to other top performer micro-computers, such as ODROID XU4 and NVIDIA Jetson Nano, we chose more lightweight but still performant Raspberry Pi 4 with 8Gb RAM [42]. It has various libraries and a broad user base, as well as enough physical interfaces for sensors. It also has an onboard WiFi card, ensures a good headless development experience, and leaves open opportunities for potential communication with an off-board computation server.

To communicate between the FC and the CC, we use MavLink's ROS package mavros [38] which will be explained in subsection 4.2.

- **USB accelerator:** Google Coral USB accelerator [26] is an Edge TPU coprocessor that can speed up local machine learning inferences. It has a USB-C interface, which is ideal to be connected to Raspberry Pi. Meanwhile, the combination of Raspberry Pi 4B and Coral gives us a relatively lightweight, powerful and sufficient solution for human tracking purpose.

- **Motors and electronic speed controller (ESC):** The decisions for motors, propellers, ESC and battery are never independent from each other. A good combination can return us efficient and stable flight. We bought miniature brushless motors which are compatible with 3-inch propellers.
  The higher the labeled current an ESC has, the better battery health and stability for the drone. However, the downside is the slightly increased weight. [37] Furthermore if the current of all electronics is lower than the allowed peak current by the ESC, no power distribution board (PDB) is needed. We chose an 35A 4-in-1 BLHeli_32 ESC which supports the DShot protocol. It is a digital protocol, therefore, ESC calibration is no longer necessary.

- **Battery and universal battery eliminator circuit (BEC):** Considering the tradeoff between battery capacity and its own weight, we went for a 1550 mAh 4S 120C LiPo. This means the battery has four cells. Each cell has 4.2V if fully charged, around 3.6V if drained but still recoverable. A 5V 3A UBEC is converting the LiPo voltage to 5V for the FC and the CC.

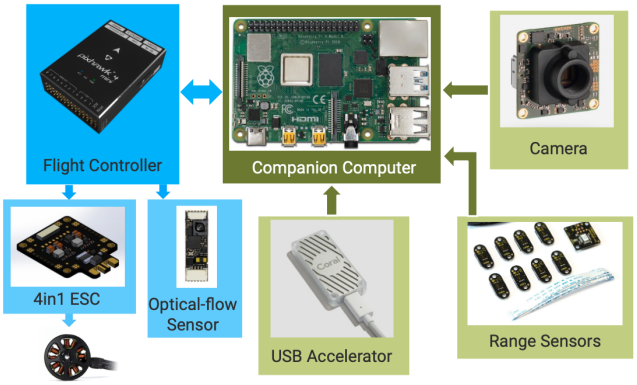


Figure 5: This graph describes how the onboard electronics are communicating with each other. On the left side with a blue background, the FC (Holybro Pixhawk 4 mini) gets signals from the optical-flow sensor (HereFlow) and the 4-in-1 ESC; on the right side with a green background, the CC (Raspberry Pi 4 Model B) is connected to the main camera, the distance sensor tower and the Google Coral USB Accelerator. The communication between the FC and the CC is succeeded via signal cables using MavLink.

3.3 Flight Test Setup

Our drones used to be tested in a flight test room at the Institute of Flight System Dynamic before COVID-19 spread. During the pandemic time, the lab usage is limited and we work at home and cooperate online.

Once the drone hovers stably in Stabilize mode without any rope constraints, we started testing AltHold mode, where we confronted another issue: the drone went out of control and flew into the ceiling once being armed. Here are some inspiring ideas on how we created a safe niche in any room for an indoor flight test.

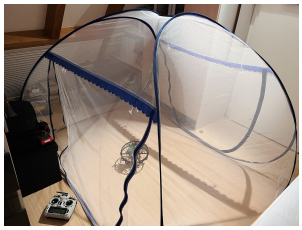
- **Tripod stands and nylon ropes:** We borrowed sandbags and tripod stands (see Figure 6a) from a theatre. Figure 6b is a close-up shot of the test drone without a cage, hanging in much more lightweight support structures than in Figure 6a. Furthermore, we reduced from four to two tripod stands to increase the degrees of freedom. It is crucial to balance the center of gravity and to level the FC before take off.
  For less powerful and more miniature drones, instead of using heavy tripod stands, ropes attached to the chairs' back would be enough.

- **Mosquito net:** Astonishingly, a mosquito net as in Figure 6c was massive enough to prevent an out-of-control lift off into the ceiling. Choose the correct type of mosquito net, which can be folded and unfolded instantly, to save time.
  However, this has not been tested for a drone without a protective cage, because the propellers might damage the net. Also for the same reason, a buffered carpet could be needed, so to protect your test drone and your precious wooden floors.

The issue was the extended Kalman filter (EKF). It used multiple sensor data sources for altitude estimation. The barometer output could be influenced by changes in air pressure, such as someone opens the door. Since the drone only flies above flat ground, we now only rely on the output from a range finder that is facing downwards.

(a) Test setup v1

(b) Test support structure v2



(c) Test protection v3

Figure 6: Left: A prototype of the flight test stand to constrain the quadcopter's flying space. Right upper corner: The second version of flight test support structure with reduced weight. Right lower corner: A mosquito net as the flight test place to limit the drone's altitude.

3.4 Calibration and Tuning

- **PID-tuning:** Every time after replacing any components onboard, especially when changing a battery with different capacities as well as different weights, the PID gains have to be re-tuned. The principle is easy: A drone shall hover stably without much oscillation but still ab able to react fast to throttle changes [52].
  Before tuning PID values, hold the middle point at two opposite sides of the main frame, see to which side the drone tends to tilt, in order to check the center of gravity (CoG). In our case, we moved flexible parts slightly to the back, so to balance the camera weight at the front.

- **Optical-flow sensor tuning:** After setting up the optical-flow sensor HereFlow according to its official site [22], we

calibrated it based on ArduPilot's tutorial [15] by scaling *OF.bodyX(Y)*, *OF.flowX(Y)* and *IMU.GyrX(Y)* till their log curves have a similar amplitude. Currently, EKF2 is used instead of EKF3.

- **Camera calibration** Calibrating a camera is in fact estimating its characteristics by defining the relationship between a point in a 3D space and the same point in a 2D image [18]. To remove the distortion effect of our fish-eye lens, regularly we run a calibration program while holding the camera in front of a black-white marker grid.

The last two calibrations above are for the sensors mentioned in subsection 4.1.

4 Autonomous Flying

All versions of Golden Snitch, including the last version GSv3, can be flown using a remote controller or fly autonomously with preset commands and a decision tree. It reacts to sensor data regarding the environment and human players.

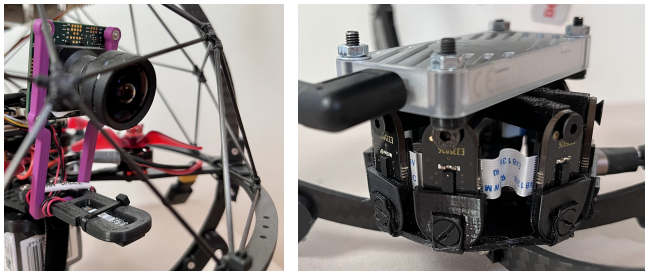
4.1 Sensors onboard

Besides sensors in the FC, we have another three kinds of sensors that are integrated to support autonomous flying.

The first one is a monocular RGB camera [31] with a 180-degree of view fish-eye lens. The drone is designed to fly between 1 and 2 meters above the ground and shall track human posture as targets, while flying with a relatively low speed compared to racing drones. Therefore, the camera is installed in the front of the upper part of the cage, facing forwards and tilted slightly downwards. To avoid occlusion by the cage rods (see Figure 7a), it is mounted as close to the cage as possible.

Since this is a drone without any off-board ground station or tracking system, that should fly in a random room, where fairly inaccurate till almost no GPS signal can be received, it needs an alternative solution. The HereFlow optical-flow sensor [22] has an optical sensor and a range finder, which is mounted below the camera facing downwards. It compares the ground pattern of one time-step with the previous one to compute the position changes and accelerations in the x- and y- directions in local space. The range finder is responsible for value and changes estimation in the z-axis in its local space.

TeraRanger Multiflex 8-sensor kit [54] includes eight range sensors connected in a chain, each sensor can detect obstacles from 0.2 m to 2 m in theory. We built them in a 3D printed tower facing 8 directions as shown in Figure 10a.



(a) Main camera and optical-flow sensor (b) Distance sensor tower

Figure 7: Left: The main camera looking out between cage rods, fixed on the top of the downward-facing optical-flow sensor, which consists of a low-resolution visual sensor and a range finder. The camera's image is slightly occluded, is corrected by algorithms. Right: Google Coral USB Accelerator and distance sensor tower sitting at the bottom of the drone cage. Their support structure is 3D printed in elastic material TPU.

The optical-flow sensor is directly connected to the FC via CAN. Whereas the camera and the distance sensor tower both have a USB interface connected to the CC with available ROS packages, then communicating with the FC via MavLink.

4.2 Software Architecture and State Machine

In Figure 8 is the simplified ROS node graph of the sensor integration. The yellow nodes are not self-written but from manufacturers and open-source projects.

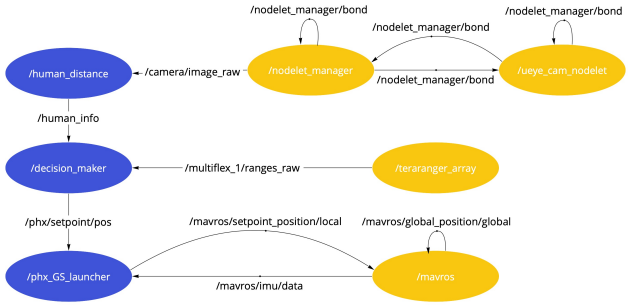


Figure 8: The software architecture of ROS nodes shows how GSv3 does sensor fusion on the CC for autonomous flight. Yellow nodes are ROS packages offered by manufacturers or open-source projects. Blue nodes are ROS packages handling sensor data. The nodes are written by Phoenix Robotics Flight Team [51] members.

The blue nodes in Figure 8 are our customized ROS nodes.

- **/phx_GS_launcher** can choose to launch needed ROS nodes which are contained in certain Python scripts at the game start, that is after the LiPo battery is connected. The default script contains a simple finite state machine (FSM), in order to prevent sending movement messages before the drone is armed and leaves the ground. Besides, the FSM always gives an option to land, no matter in which state the drone currently is in.
  The launcher builds connection from the CC to the FC via MavLink. Here it is assumed, that all commands sent to the FC go through this node.

- **/human_distance** reads the camera topic /camera/image_raw, which is the default topic of /ueye_cam_nodelet with a nodelet manager. The raw image information is then analyzed to detect a human player and estimate the distance and relative position from the camera to the target. This is described in subsection 4.4 in detail. The result, the distance and the angle, that reflects the direction of the target, are written into human_info channel of this node.
  This node's detection range is limited due to the camera lens' field of view (FOV) and based on the chosen human pose detection algorithm. In our case, it is a range of about 6 meters with an adjustable FOV of 130 to 220 degrees.
  Here it is assumed that the camera node is publishing constantly and regularly, meanwhile, the coral USB accelerator is attached and initialized.

- **/decision_maker** enables a simple decision tree to control the drone's behavior at a high level. This means, this node gathers information of subscribed topics sent from the nodes /human_distance and /teraranger_array. (The node /teraranger_array is offered by the manufacturer of our distance sensors.) Then send its decision, such as "fly forward for one second" to the node /phx_GS_launcher. From there a corresponding mavros message will be sent to the FC, there the

high-level behavior command will be translated into a low-level motor command.
This node assumes that the sensor data input stream is consistent. Furthermore, it only makes decisions for behavior changes which are referred to the drone's local space. Its previous decisions and the history of sensor readings are not considered. For instance, a detected obstacle or a tracked human target is not memorized. How a decision is made is described in subsection 4.5.

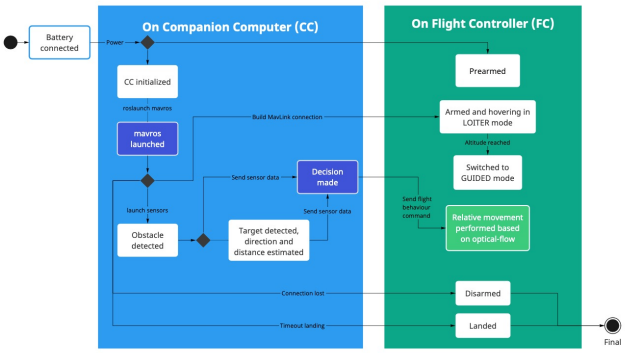


Figure 9: The graph of GSv3's states in a tangible AR game. Similar to the electronics layout in Figure 5, blue represents states in the CC and green represents states in the FC.

After connecting the LiPo battery, it powers and initializes the FC and the CC. All commands from the CC to the FC should be sent through the /phx_GS_launcher node as shown in Figure 8 via MavLink. These commands include when the FC should arm and take off as well as how the FC should move the drone relatively in its local space regarding the decision-making process on the CC.

ArduPilot's open-source framework ArduCopter for copters supports plenty of flight modes [12]. We are using LOITER mode to take-off and to hold the flight height (Z-coordinate) as well as the xy-position relatively to the ground by using an optical-flow sensor instead of a GPS receiver. After reaching a preset height, it switches to GUIDED mode, where the FC reads waypoints in local space sent by the CC and flies the points.

The drone will be disarmed, once the MavLink connection no longer exists. It will perform a landing in place, when a preset timeout is done. This is set accordingly to the flight tests made previously with the corresponding LiPo batteries.

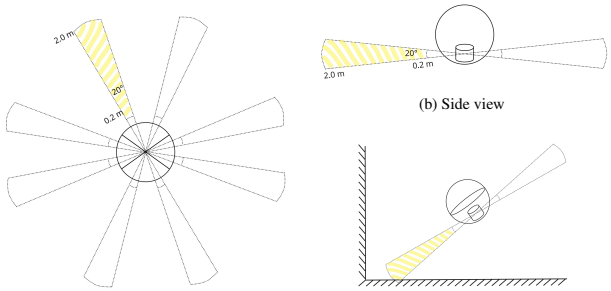
Be cautious, all the CC commands are assumed to be accepted by the FC. The CC does not check or get feedback on whether the FC executes those commands and changes its states or not. This might lead to inconsistency of commands on the CC and the FC.

4.3 Obstacle Avoidance

The manufacturer of our distance sensors, Terabee [54], provides a ROS package [53] to run its sensors. It is convenient to control the sensor tower as an array.

Each sensor has a 20-degree FOV and can detect objects appearing within the range from 0.2 to 2 meters away. To prevent any possible occlusion caused by the vertical outer skeleton rings, the eight distance sensors set at the bottom of the drone are not evenly facing outwards around the drone as shown in Figure 10a.

However, when the drone is not leveled, especially as it is moving fast, false positive (see Figure 10c) detection occurs, as if the ground was also an obstacle.



(a) Bird view

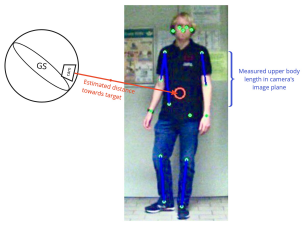
(b) Side view

(c) Case of false positive detection

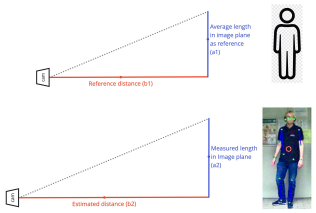
Figure 10: The concept and range of obstacle detection using a distance sensor tower.

4.4 Target Detection and Distance Estimation

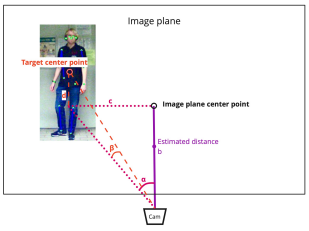
The input raw image is sharp as long as the target is within a range of 6 meters from the camera. A 220° fish-eye lens is mounted on the camera. However, the area within an FOV of 130° to 180°. As mentioned in subsection 4.1, occlusions because of the inner cage are also taken care.



(a) Body length measurement in image plane



(b) Distance estimation from camera towards the closest player



(c) Direction estimation from camera towards the closest player

Figure 11: Human pose detection based on PoseNet [10] using Coral [21]. Its purpose is to estimate the distance from the drone towards single or multiple target people, by comparing a standard body size (a human body of 1.75 m) with the detected human pose.

The desired output includes an estimated distance from the detected target to the drone b_2 and two angles α and β indicating the relative position of the target towards the camera.

The length of a body part in the camera view, in our case the upper body length, a_2 is calculated by differentiating two end points' 2D coordinates on the image plane. We measured in advance the height of a reference person (1.75 m) and his upper body length in the image a_1 as well as the distance from the camera to the image plane center point b_1 at the same time. With the known input $\frac{a_1}{b_1}$ and a_2 we assume the ratio $\frac{a_2}{b_2}$ is the same as the input and therefore we

gain b_2 from

$$\frac{a_1}{b_1} = \frac{a_2}{b_2} \rightarrow b_2 = \frac{a_2 \cdot b_1}{a_1}$$

as shown in Figure 11b.

Obviously, the estimated distance b_2 will not always match the reality, since not every player will have the same body length as the reference person. We are searching for a better solution to get the distance towards a human target for a more accurate estimation.

Furthermore, we calculate the relative position of the target by measuring the coordinate of his center point in the image, as the red circle shown in Figure 11c. Assume the image center point is at $(0,0)$ on the image plane, with the known input including target coordinates (c,d) and the calculated distance b ($b2$ from the last step), we gain the desired angles α and β as the followings:

$$\sin\alpha = \frac{c}{b} \rightarrow \alpha = \arccos(\frac{c}{b})$$

$$\sin\beta = \frac{d}{\cos\alpha \cdot b} \rightarrow \beta = \arccos(\frac{d}{\cos\alpha \cdot b})$$

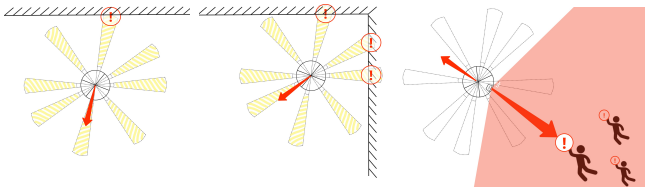
A fast movement can result in blurry raw images, therefore reducing the vibration of the drone flight is crucial. So far we have not applied extra algorithms to eliminate the blurring effect.

4.5 Decision Making

Since the drone is not aware of the environment, namely the global space, with its current setup, its decision making is limited to reactions towards sensor signals in its local space. In another word, it can fly to a relative position in its local space, for example, flying two meters or for three seconds straight forward, which means that it moves towards the direction in which its front side is facing. To simplify the environment, we set a flight height at 1.5 meters, which can be optimized and adapted to a specific game rules. Therefore, besides takeoff and landing, the drone mainly keeps its ground altitude at the given flight height, which is in default at 2 m.

Our script has covered several circumstances regarding the input of the distance sensor tower and the camera. Meanwhile, with help of the optical-flow sensor, the drone reacts to these circumstances. There are two main cases, the first case is when a wall or another static obstacle is detected, the second one is when a human player is spotted (see Figure 12). The first case is taken as the first priority for now.

Once the drone has decided the next destination, it rotates till its camera is facing to the desired moving direction.



(a) Priority one: obstacle avoidance detected by single sensor

(b) Priority one: obstacle avoidance detected by multiple sensors

(c) Priority two: player interaction

Figure 12: Corresponding actions under specific circumstances. Left: In the local space of a drone, if one distance sensor detected obstacles, the drone flies in the opposite direction of the sensor's position, till the signal disappears. Middle: If more sensors reported obstacles, the drone flies the to the opposite direction of the mid-position of all sensors that have received signals. Right: If the camera detected human players, the drone can either fly towards the closest player or away from him, depending on the game rules.

4.6 Simulation and Testing

During the deployment, flight tests got blocked very often. To prevent any time lost due to crashes or waiting for replacements being shipped and batteries being charged and also to accelerate future ML training for the drone's behavior, a simulation as a digital-twin test playground is needed.

ArduPilot naturally supports its own open-source simulator Simulation-in-the-Loop (SITL) [13] running in gazebo, which also uses mavros [38] to exchange data between sensors and main computers, just like on a real drone. ArduPilot SITL helps verify our obstacle avoidance solution by defining the environment and the sensor range. Furthermore, since the global environment is on the database and theoretically unknown to the drone agent, the path-finding algorithms can be trained and tested in this simulation, too.

5 In-Game Interaction

5.1 Our Superhuman Sports Game's Concept

The paper "Catching the Drone" [24] designed a SSG, where two teams compete with each other on a playing field that is sectioned in 3 zones. Their objective is to score points by throwing an autonomous flying caged drone through goal rings. There are three goal rings on each end of the playing field.

Using a drone as a game ball opens up new possibilities for the game mechanics, especially giving the players a tangible experience along with the AR displays. The players are able to execute multiple abilities along with the drone.

5.2 Requirements Referred to the Game Rules

The game rules require a drone, that flies close to players, that can even be caught without hurting them. It is also desired, that catches and goals can be detected. The drone should be crash-resistant, so a throw motion would not lead directly to the game's end. Furthermore, the drone needs to be tracked in players' AR headsets.

5.3 Touch Sensing

5.3.1 Requirements for Touch Sensing

The touch sensor or sensors have to fulfill the following requirements:

- **Low weight and size:** It is important for the copter's performance and stamina to keep the drone as light as possible. As the touch sensors will be mounted on the drone, we have to choose a light sensor while covering maximum space at the same time.
- **Not too many additional cables:** Additional cable could potentially complicate further adjustments. So keeping it simple is desired.
- **Not sensitive to vibration (false-positives):** The drone will naturally have little vibrations through the spinning propellers or fast and sudden movements.
- **Only react to human touch:** Ideally only human touch will be counted as a touch. Non-human touches could be collision against other object.

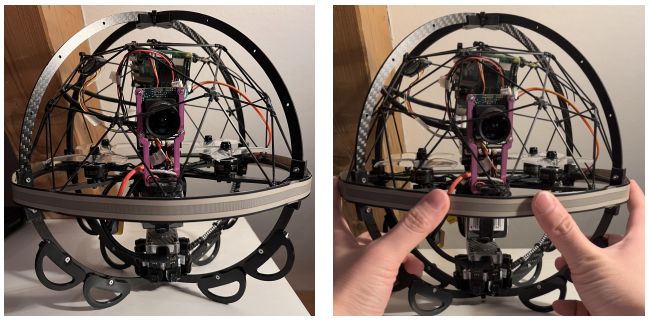
5.3.2 Materials

Force Sensing Resistor (FSR) sensors are inexpensive and have a wide variety of sizes and shapes. Resistance change drastically depending on the amount of pressure. Because the resistance changes dramatically at first, but almost stagnate afterwards, it is not suited for measuring a linear scale. However, for the simple detection such as, if pressure is existing or not, FSR sensors are an excellent choice.

For touch sensing we decided to use Interlink Electronics' *Interlink 408 Extra-long force-sensitive resistor* [2], which is a 60,96 cm long and 1.53 cm wide FSR sensor. The FSR sensor Interlink 408 is lightweight and can cover a generous amount of surface. It will not burden the drone with its weight. Moreover, it is very effective in sensing light pressure and is not susceptible to vibrations that serves to avoid false-positive touch feedback. FSR sensor have their drawbacks [3]: They could be pre-loaded through bending. That results in a diminished dynamic range or resistance drift. The force sensing part of the sensor (active area) is very fragile. Dents, kinks and creases will trigger false results. Temperature and excessive humidity also affect the results as [4]. Generally, the FSR's resistance is growing with a higher ambient temperature.

5.3.3 Touch Sensing with Force Sensitive Resistor (FSR)

The drone needs to know, when it is hold by a player to adapt its flying pattern. For this purpose, we wrapped the FSR 408 around the drone's equator. When someone catches the drone, the FSR sensor detects pressure on its active area and sends the message "Under Pressure" to the connected Raspberry Pi Model 4b. To avoid false-positives, we only count the detected touch as human touch, if the sensor is "Under Pressure" for 10 *counts*. In that case the message "Quadcopter is being hold" will show and the drone now knows that it is being held by someone. 1 count takes approximately 1 sec and is the variable *counter* in the code *touch_detector_4.py*. The count will increase, only when the FSR continuously senses pressure. Once there is no pressure, the *count* will be reset to 0. In the other cases we assume no touch or it was an unintentional touch, like a crash against the wall, other objects or accidental bumping into player on the playing field. In those cases the message will be "short wall or human contact" or "No Pressure".



(a) FSR touch sensor on the drone's equator

(b) Holding the drone on the FSR touch sensor with both hands

Figure 13: FSR touch sensing

So far, the FSR sensor is attached to the middle ring of GS using a strong double-sided tape. This is because no pressure such as zip-tie should be applied on its surface.

5.4 Goal Construction

5.4.1 Requirements for the Goal Construction

The goal has to fulfill the following requirements:

- **Crash resistant:** The goal has to be crash-resistant, because the drone could hit it while flying through or pass it many times. Upon a crash the goal should be still standing and working.
- **Appropriate height:** The drone has a diameter of about 25cm. Therefore the inner diameter of the goal should be at least 40 cm. The goal should be about 2 to 3 meters high, so that it is slightly above a player's head. They have to throw the drone to score a point or jump to block a goal.
- **Appropriate size:** The drone has a diameter of approximately 25 cm. The goal should leave enough space to score a goal, while not making it too easy for the players.
- **Goal detection and count:** When the drone flies through the goal ring, the score should be detected reliably. In addition, the score should be counted and increased automatically.

5.4.2 Materials, Sensor and Technical Specifications

Formed on the requirements above, we decided to built the goal of the following materials and items:

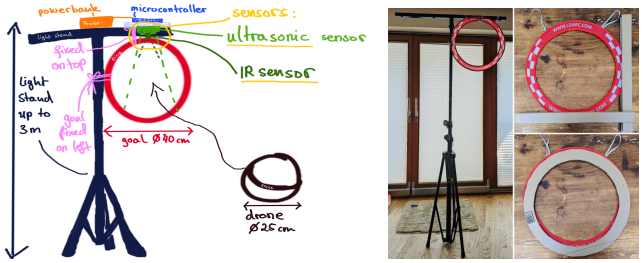
- **Goal gate ring:** The goal is a flexible, crash-resistant and drone-friendly gate ring which is designed for FPV drone racing. The ring we use has an outer diameter of 50 cm and an inner diameter of approximately 40 cm. That leaves about 40 - 25 = 15 cm around the drone to get through the goal ring [7].
- **Stand:** The stand is where the goal ring is attached. For this purpose we chose a light stand, which is extendable up to 3 m and stands on a very stable mammoth foot stand design [8].
- **IR sensor:** The *ARD LINE FINDER2 Arduino (ST1081)* is an adjustable infrared light sensor that can detect obstacle in distances up to 40 cm like the goal's inner diameter (40cm) [1]
- **Ultrasonic sensor** The ultrasonic sensor *HC-SR04* is an accurate distance sensor and detects obstacles within 1 cm and 3 m [5].
- **Microcontroller** We use the Particle Argon as our microcontroller that can connect and publish events to other devices wirelessly and effortlessly [6].
- **Gate Stabilizer:** We designed a gate ring stabilizer out of cardboard boxes, in case the goal is flopping around too much in our tests. In further evaluations we will find out, if a stabilizer is necessary or not, see Figures 16b, 14b and right side of Figure 15b.

(a) Gate stabilizer cardboard cutouts

(b) Gate stabilizer in a interleaving square shape

Figure 14: Gate reinforcement

5.4.3 Final Goal and Detection



(a) Goal construction sketch

(b) Reinforced gate and gate on a 2m stand.

Figure 15: Goal construction on a tripod stand.

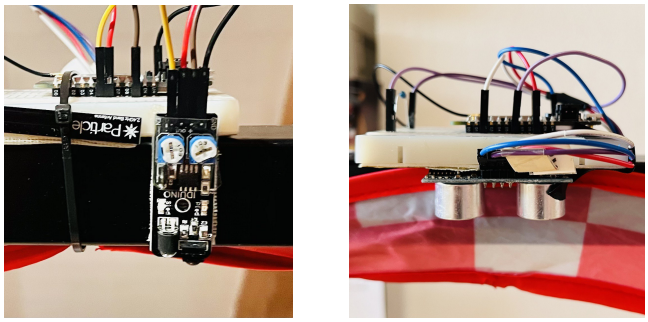
The final goal has the FPV gate ring attached to the extendable light stand, see left side of Figure 15b. On the top of the gate ring an IR sensor or an ultrasonic sensor with the microcontroller and a power bank is mounted. The sensors are facing top-down to detect the drone. As our power supply for the Particle Argon we used a mobile, medium size power bank that should have enough power to provide for a game of at least 30 minutes.

After researching about different sensor types (infrared, magnetic, capacitive, inductive, ultrasonic, radar), we chose to try out the two simplest and affordable ones for the goal detection: an infrared light sensor and an ultrasonic sensor, see Figures 16a and 16b.

Obstacle Detection Sensors	
Infrared IR	**Ultrasonic**
Advantages	**Advantages**
· sensor is cheaper · sensor is smaller · detection is faster	· color independent · easy distance calculation · unwanted interference is not that likely
Disadvantages	**Disadvantages**
· not good with infrared absorbent materials (eg. matte objects) · color dependent (eg. black is not good) · interference with various IR sources (eg. sunlight)	· not good with sound absorbent materials (eg. foam) · detection is slower

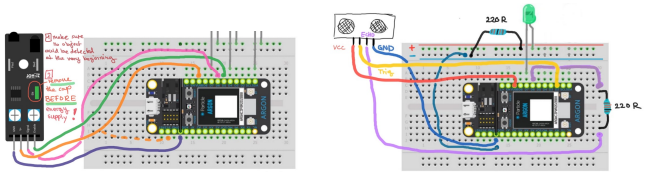
The **IR sensor** sends the message ”Hindernis erkannt! Tor :D ” and the count of the score, if it detects the drone. The counter will only increase, when there was a LOW signal in between to make sure that the same goal will not be counted twice.

The **ultrasonic sensor** works likewise, except it sends the message ”Goal” plus the count of the score.



(a) IR sensor (b) Ultrasonic sensor

Figure 16: Goal detection sensors attached to the tripod stand.

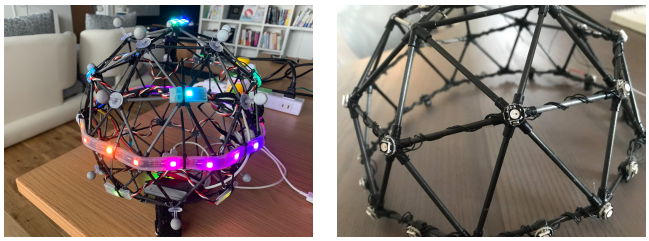


(a) IR sensor (b) Ultrasonic sensor

Figure 17: Connection graph of sensors for goal detection.

5.5 LED’s Installation for Tracking

Figure 18 shows two generations of LED on the cage for GSv1 and GSv2. As mentioned in paper [24], the LED pattern is used for AR tracking in the game. A further update of LEDs on GSv3 will be made (see subsection 7.4).



(a) LED cage v1 (b) LED cage v2

Figure 18: LED cage prototypes. Left: The first prototype of LEDs on the cage. However, the transparent protection wrapper cannot be glued onto the carbon fiber cage, fixing with zip-ties resulted in vibration and led to unreliable LED tracking later on. Right: The upper half cage with embedded LEDs in its connectors. This concept helps to have a much more stable LED detection result.

6 Evaluation

6.1 Evaluation on Crash Resistance

The PLA connectors on GSv1 were so fragile, that with every crash after a flight test one or two connectors were broken.

Now on GSv3 with the outer skeleton and the elastic TPU connectors absorbing the impact, the cage barely breaks when falling from about 1 m above the ground.

After at least 20 flight tests, only one carbon fiber rod on the inner cage broke and needed to be replaced. This is caused by replacing the thicker rods (3 mm diameter) using the thinner one (1.5 mm diameter), which makes the inner cage more lightweight but fragile. However, it was directly in front of the camera. This points out that the camera needs a better protection and a better fixing solution – a little closer to the center and further away from the cage.

6.2 Evaluation on Obstacle Avoidance Detection

Here the detection range of different static obstacles are listed. Here we observe the closest sensor’s output while approaching the testing obstacle. We hold the drone and start from 2 meters away, stop when range finders are not displaying ”inf”.

Then we take the average measurement of 10 trials in each case approximately. The tests are made with the disarmed drone, so the sensors do not rotate or translate. Moreover, a timeout clearance setting is necessary, since the data input stream can be noisy. A too long timeout might result in poor and slow response.

Obstacle types	Average detection range
White walls	1.954 m
White corners	1.892 m
Wooden shelves	1.843 m
Transparent windows	1.545 m
Black curtains	1.056 m

Detecting darker materials has in general a slower response and less detection range. These tests are made at the evening with artificial lights on. How clean and transparent a window is could also influence the testing results.

As shown in Figure 10, the distance sensor tower is not covering all 360 degrees of its environment. Therefore, obstacles like humans, especially when they are slightly moving, the positive detection time

window will be very short. Depending on the game rules, setting a different clearance timeout adjusts the positive detection rate of non-static, thinner obstacles.

6.3 Evaluation on Target Tracking

The following table shows various cases, each cell represents one special case as described in the table's header column and header row. It shows how often human targets within the maximum detection range (6 meters) get recognized. For each case we have done 5 tests in a same room with different light conditions. "Multiple targets" here indicate two testers with facial masks who have an approximately average body height. We count a target as "detected" when the range output is not "inf" for at least one second.

	Single target	Multiple targets
Static	100%	90%
Slow movement	100%	95%
Fast movement	70%	55%
Occluded	70%	60%
Truncated	50%	45%
Side view	60%	50%

The occlusion caused by the inner and outer cage should be eliminated by algorithms. But we think, to solve this issue physically would give more reliable results.

6.4 Evaluation on Goal Detection

6.4.1 Goal Detection *without Drone - IR Sensor*

The IR sensor with enable pin is able to detect a white sheet of paper in up to 30 cm to 40 cm under good conditions. Good conditions means that the tests where conducted in a dark room without the presence of sunlight to avoid irritations or in a room with artificial light. The white paper was place as perpendicular as possible to the sensor to get the maximum response from the sensor.

6.4.2 Goal Detection *with Drone - IR Sensor*

Until the end of this project, we have not designed how the drone flies to a gate by itself. This is also mention in subsection 7.3, that a throwing motion can be applied in the future. So far, all goal detection tests are made in a way, that either the drone is controlled in Stabilize mode or the armed, spinning drone held in one hand.

We encountered the problem that the IR sensor is color and material dependent (eg. black-matte object will not reflect much infrared light, hence it will not be detected) very early on. All or most parts of our 3D printed cage and drone are black, which probably absorbs the IR light. Only the raspberry pi's board is reflecting sensor signals reliably, because it has a green flat surface. Unlike the board, the other spaces are sparse, dark and curved. The drone does not provide much surfaces that could reflect the infrared light. The reflective surface should be ideally perpendicular to the sensor, otherwise the detection range will be diminished.

In the table below, we test the normal cases, where the respective team will score 1 point, when the drone flies through the goal ring. The cases are: the drone could fly through the center of the ring or off-centered more right or left (horizontally) and top or bottom (vertically). During the tests the IR sensor was set to detect a white paper within 46 cm and a hand in 25 cm, detecting the drone was also about within 25 cm. The IR sensor tests were conducted in a dark room with artificial light.

Normal Tests - Drone flies through the goal		
Centered O	Off-center horizontally —	Off-center vertically ||
detected reliably	detected	detected

For the false-positive cases, the drone might fly against (but not through) the goal from underneath, above and the sides. The goal ring might flop around and bend after it gets hit, causing unwanted detection. In this way the drone could be somehow detected, although it did not go through the goal.

False-positive - Drone flies past the goal			
front	under	on the left	on the right
not detected	not detected	not detected	not detected

In the false-positive tests for IR sensor, no false-positive goals were detected. Although the drone hit the ring underneath on the bottom and once from the right side, no goal was detected. On the left side, the drone cannot hit the ring, because the stand in its way. When the drone was just in front of the goal, there were also no false-positives detected, as light spreads in a straight line in the direction, where the IR lamp is pointing (in our case it points at the bottom direct).

6.4.3 Goal Detection *with Drone - Ultrasonic Sensor*

We carried out the same test cases with the ultrasonic sensor. The ultrasound sensor works better on the drone, since it is color independent. So a goal is detected more safely with the detection range written in the code. However, the ultrasonic sensor seems to be more unstable, when it comes to not detecting the same goal several times. This ended in many false-positive goals with the ultrasonic sensor. The ultrasonic sensor tends to go HIGH and LOW again in between, while the drone goes through the goal ring. That resulted in false-positives, as the goal counter increases, whenever there was a switch from HIGH to LOW and from LOW to HIGH again. During the tests the ultrasonic sensor was set to detect goal within 30 cm.

Normal tests - Drone flies through Goal		
Centered o	Off-center horizontally —	Off-center vertically ||
detected reliably	detected reliably	detected reliably

False-positives - Drone flies past Goal			
front	under	on the left	on the right
detected	not detected	not detected	not detected

The IR sensor seems to be more stable, when it comes to not having false-positives (if it is the same goal), for the ultrasonic sensor that was not the case. Moreover, ultrasonic waves spreads uniformly in all directions, so a goal was already detected in front of the goal ring , before the drone has flown through it.

6.4.4 Goal Detection General

If the drone flies against the IR/ ultrasonic sensor, we have currently no protection. A net around the goal like those in soccer, could prevent false-positives by stopping the drone from going further or/and stabilizing the goal, if it gets hit. To avoid players touching the goal, we could draw a line in front of the goal, leaving enough space in between or place the goal far behind the playground.

6.5 Evaluation on Touch Detection

6.5.1 Touch Sensor without Drone

Depending on the installed resistor between GND and pin 7 of the raspberry pi, the sensitivity of the FSR sensor changes. For the tests of the FSR senor's sensitivity we tried out various resistor from 100 R to 10 M Ohm. Vibrations, bending and other possible noise was manually simulated by shaking hand motions.

Sensitivity Table		
Resistor	Touch behaviour	Noise behaviour
100 R	nothing happened	nothing happened
560 R	had to press hard to get a reaction	no noise error
330 R	had to press hard to get a reaction	no noise error
4.7 K	good response	not sensible to noise
1 K	good response	not sensible to noise
2 K	good response	not sensible to noise
10 K	better response	little bit sensible to noise
20 K	better response	little bit sensible to noise
51 K	even better response	little bit more sensible to noise
220 K	nothing happened	nothing happened
1 M	sensitive response	very sensitive to noise
10 M	sensitive response	very sensitive to noise

6.5.2 Touch sensors on Drone

We mounted the FSR sensor with a 5.6 K resistor on drone, because using this resistor turned out to be not overly sensible to noise, while having sufficient response, when the drone was hold.

For the normal tests, we want to see, if a catch is reliably recognized by the drone. The drone could be hold lightly, normally or caught in midair.

Normal Touch Tests		
Light hold pressure	Hold pressure	Catch pressure
detected occasionally	detected reliably	detected reliably

Depending on the holding point, on which the total weight of the drone is sitting, the light hold is occasionally detected. If the main gripping points are under the middle ring, then the touch sensor is not capturing enough force, so the catch cannot be recorded reliably.

False-positives could appear, because of noise (e.g. from the copter's movements), if the drone flies against a wall, other objects or against a player. For those cases, we do not want to consider them as "being hold" nor caught. For testing, we said that "holding" occur, when the FSR sensor detects pressure for 10 counts. If the count is less than 10, the detected pressure, will not be counted as a hold/ catch.

False-Positive Touch Tests		
Copter noise	Touch against wall	Touch against people
none	detected occasionally	barely detected

With well-tuned PID gains, the vibration of the drone is low. Even With high vibrations, the touch sensor does not report a false-positive detection either.

Depending on the flying speed and the contact surface of a crash, either into walls or into a person, the touch sensor detects the a hold/ catch, if the contact prevails for more than 10 counts.

Setting a timeout of 10 counts, false-positive detection barely appears. The longer the timeout we set, the smaller the chance that a false-positive case get detected. However, it is a trade-off between detecting almost solely real catches and maintaining the game flow. The current timeout count of 10 is probably too long for the the game. We have to find a reasonable timeout count to reliably detect catches and not accidental touches but, at the same time, the timeout count should not be too long, so that the game stagnates.

7 Future Work

7.1 Localization and Mapping

The current concept only allows us to set waypoints in the local space of the drone or to send attitude commands for the next movement of GS. To gain the perception of the drone's global position, we'd need appropriate sensors, computers and algorithms for localization and mapping.

Two possible methods could be applied. The first is server-based, so the drone does not have to carry heavy powerful sensors or computers. The onboard computers communicate with the server either via WiFi or using Bluetooth. Because decision-making shall not hesitate in some urgent cases, a stable and real-time connection is necessary. The second one is a simplified onboard approach. Before starting the game in a new environment, one can hold the drone and do a one-time mapping manually. There the power supply or the needed time is not a constraint anymore. Then the drone only has to do localization in the game.

In the future, either with a bigger drone that can undertake more weight onboard or with new miniature but powerful sensors such as stereo or depth cameras, more complicated algorithms such as Simultaneous Localization and Mapping (SLAM) and trajectory planning can be applied.

7.2 Flight Time Optimization

The flight time is limited by the battery capacity, the engines and propellers' lift, the FC and the CC's power consumption during flights, for instance if its PID gains are well-tuned, and the total weight on board.

With all the constraints, we would like to have a longer flight time for a continuous game, as well as a more agile and lightweight drone-ball. There are several points that can be improved, such as adding ducts back and rearranging the CC's position to optimize the airflow, to increase the lift and to decrease the noise level. Also, we can replace some parts of the current combination of ESC, motors and propellers, to find the most efficient power consumption solution.

7.3 Drone Behavior after a Catch

Desired is to have the drone reacting to a catch in the game. The motors should slow down but not yet stop spinning, once the drone is caught with spinning propellers. A catch is detected by the touch sensor mentioned in subsection 5.3.

To not interrupt the game flow, after being caught, the drone shall be able to take off again. ArduPilot offers a flight mode called Throw mode [16]. It could be possible to have propellers spinning more with a throw motion. However, Throw mode does not seem safe and promising in an indoor environment yet. Instead, a more appropriate game design might be more reliable.

7.4 LED Tracking on the Current Cage

In GSv3, after the removal of the lower half cage, due to the time constraint and the purpose not to occlude any sensor range, we want to add the LEDs wrapped in 3D printed soft cover, then bind them onto the outer skeleton where no sensors are facing towards, instead of being embedded in the inner-cage connectors.

8 Conclusion

In this paper, we depicted how an autonomously indoor flying drone with various constraints and requirements is built and developed. It shall move away from close obstacles and fly towards detected human targets or behave correspondingly to specified game rules. It needs to hover long enough to finish a game. It has to be safe and small enough just as a normal volleyball that can be caught by hand. Along with that, the drone has to be able to recognize, when it itself is being hold or caught by a player. With these objectives, we made many compromises and also tried hard to balance trade-offs, in order to support a playable drone-ball SSG with AR in the near future. We

improved many designs and replaced various parts after each crash in flight tests, such as the cage and the sensor tower. Moreover, we built a goal for the scoring system of the game.

Last but not least, we believe there is still a big room for optimization and more flight tests are necessary. To handle noisy data for obstacles and target detection, the altitude and relative position estimation, further parameter tuning and methods such as EKF and Fast Fourier Transformation (FFT) can be used to reduce such noises.

During this project, we learned how to build things from scratch based on numerous researches, we became more patient about dealing with failures. We are looking forward to optimize the drone and its in-game infrastructure for a playable game.

Acknowledgments

We sincerely appreciate the student group Phoenix Robotics Autonomous Flight team [51], especially Henrik Dobbe Flemmen, Kevin Röhrborn, Dominik Weiß, Zichong Li, Martin Zimmermann, Kai Eberl, for your contribution and creativity in the last two years.

We would like to thank the Research group Augmented Reality [34], the Chair of Automatic Control [27] and the Institute of Flight System Dynamics [30] at the Technical University of Munich (TUM) for your effort, support and instructions.

We are also very thankful to our families and friends, who helped us during the Corona time, especially for bearing the drone noise and offering extra free space for testing, equipment and materials.

References

[1] Ard line finder2 arduino - ir-hindernissensor. https://www.reichelt.de/arduino-ir-hindernissensor-2-40-cm-38-khz-ard-line-finder2-p282521.html?CCOUNTRY=445&LANGUAGE=de&trstct=pos_15&nbc=1&&r=1, visited on 16.03.2022.

[2] Extra-long force-sensitive resistor (fsr) - interlink 408. https://www.adafruit.com/product/1071, visited on 16.03.2022.

[3] Extra-long force-sensitive resistor (fsr) - interlink 408 - datasheet. https://cdn-shop.adafruit.com/datasheets/FSR400Series_PD.pdf, visited on 28.03.2022.

[4] Fsr 101 - force sensing resistor theory and applications. https://www.sensitronics.com/pdf/Sensitronics_FSR_101.pdf, visited on 28.03.2022.

[5] Hc-sr04. https://www.mikrocontroller-elektronik.de/ultraschallsensor-hc-sr04/, visited on 28.03.2022.

[6] Particle argon. https://docs.particle.io/argon/, visited on 28.03.2022.

[7] Racing gate for rc drone. https://www.amazon.de/-/en/Racing-Drone-500mm-Competition-Accessories/dp/B07RHZLKCL/ref=sr_1_23?keywords=fliegendes+renntor&qid=1637935021&sr=8-23, visited on 16.03.2022.

[8] Showtec metal medium licht stativ mammoth stands. https://www.elcotec-electronic.de/Buehnen-Traverse/Stative-Hardware/Lichtstative-Lifte/Handauszugsstative/Showtec-Metal-Medium-Licht-Stativ-Mammoth-Stands::14867.html?MODsid=d29c416d1fe5c314c783c25fae0a427f, visited on 16.03.2022.

[9] Wardatul Hayat Adnan and Mohd Fadly Khamis. Drone use in military and civilian application: Risk to national security. *Journal of Media and Information Warfare Vol*, 15(1):60–70, 2022.

[10] Roberto Cipolla Alex Kendall, Matthew Grimes. Posenet: A convolutional network for real-time 6-dof camera relocalization. https://arxiv.org/abs/1505.07427, visited on 21.02.2022.

[11] Parrot AR.Drone. New game: Ar.rescue 2 *free app*. https://www.youtube.com/watch?v=w1PzPqgDHzA, visited on 09.03.2022.

[12] ArduPilot. Ardupilot arducopter documentation: Flight modes. https://ardupilot.org/copter/docs/flight-modes.html, visited on 21.02.2022.

[13] ArduPilot. Ardupilot simulation-in-the-loop (sitl). https://ardupilot.org/dev/docs/sitl-simulator-software-in-the-loop.html, visited on 20.02.2022.

[14] ArduPilot. Ardupilot, versatile, trusted, open. https://ardupilot.org, visited on 21.02.2022.

[15] ArduPilot. Optical flow sensor testing and setup. https://docs.cubepilot.org/user-guides/flow-senor/here-flow, visited on 20.03.2022.

[16] ArduPilot. Throw mode. https://ardupilot.org/copter/docs/throw-mode.html#throw-mode, visited on 15.03.2022.

[17] Mehmet Aydın Baytaş, Damla Çay, Evren Yantac, Mohammad Obaid, Asım Yantaç, and Yuchong Zhang. The design of social drones: A review of studies on autonomous flyers in inhabited environments. 05 2019.

[18] Dulari Bhatt. A comprehensive guide for camera calibration in computer vision. https://www.analyticsvidhya.com/blog/2021/10/a-comprehensive-guide-for-camera-calibration-in-computer-vision/, visited on 26.03.2022.

[19] China-Gadgets.de. DafÜr 30€? flynova pro: Der boomerang 2.0. https://www.youtube.com/watch?v=7mh66pOmkGQ, visited on 09.03.2022.

[20] Ashley Colley, Lasse Virtanen, Pascal Knierim, and Jonna Häkkilä. Investigating drone motion as pedestrian guidance. pages 143–150, 11 2017.

[21] Google Coral. Coral posenet. https://github.com/google-coral/project-posenet, visited on 21.02.2022.

[22] CubePilot. Here flow optical flow sensor. https://ardupilot.org/copter/docs/common-optical-flow-sensor-setup.html#common-optical-flow-sensor-setup, visited on 20.03.2022.

[23] TU Delft. Superhuman sports design challenge – tu delft. https://superhuman-sports.org/delft/, visited on 05.02.2022.

[24] Christian Eichhorn, Adnane Jadid, David A. Plecher, Sandro Weber, Gudrun Klinker, and Yuta Itoh. Catching the drone – a tangible augmented reality game in superhuman sports, 2019.

[25] Flyability. Flyability. https://www.flyability.com, visited on 05.02.2022.

[26] Google. Usb accelerator. https://coral.ai/products/accelerator/, visited on 20.02.2022.

[27] Prof. Dr.-Ing. habil. Boris Lohmann. Chair of automatic control. https://www.epc.ed.tum.de/en/rt/home/, visited on 20.02.2022.

[28] HobbyDirekt. Sweeper set droneball white graupner. https://www.hobbydirekt.de/Neuheiten-2021/Neuheiten-2019/Graupner/SWEEPER-Set-RTF-Droneball-white-Graupner-16580-RTF::360781.html, visited on 07.03.2022.

[29] Holybro. Pixhawk4 mini. http://www.holybro.com/product/pixhawk4-mini/, visited on 20.02.2022.

[30] Prof. Dr.-Ing. Florian Holzapfel. Institute of flight system dynamics. https://www.fsd.ed.tum.de, visited on 20.02.2022.

[31] IDS-Imaging. Ids-imaging u3-3241le-c-hq. https://en.ids-imaging.com/download-details/AB00561.html, visited on 20.02.2022.

[32] Bojana Ivosevic, Yong-Gu Han, Youngho Cho, and Ohseok Kwon. The use of conservation drones in ecology and wildlife research. *Journal of Ecology and Environment*, 38:113–118, 02 2015.

[33] Marie Kegeleers, Shivam Miglani, Gijs M.W. Reichert, Nestor Z. Salamon, J. Timothy Balint, Stephan G. Lukosch, and Rafael Bidarra. Star: Superhuman-training-in-ar. Paper: https://graphics.tudelft.nl/Publications-new/2018/KMRSBLB18/a7-kegeleers.pdf, Github repository: https://github.com/Shivam-Miglani/STAR-Superhuman-Training-in-AR, visited on 26.03.2022.

[34] Prof. Gudrun Klinker. Research group augmented reality. https://www.in.tum.de/far/startseite/, visited on 20.02.2022.

[35] Pascal Knierim, Thomas Kosch, Valentin Schwind, Markus Funk, Francisco Kiss, Stefan Schneegaß, and Niels Henze. Tactile drones - providing immersive tactile feedback in virtual reality through quadcopters. pages 433–436, 05 2017.

[36] Joseph La Delfa, Mehmet Aydın Baytaş, Hazel Ngari, Rakesh Patibanda, Rohit Khot, and Florian Mueller. Drone chi: Somaesthetic human-drone interaction. 04 2020.

[37] Oscar Liang. Fpv drone esc buyer's guide. https://oscarliang.com/choose-esc-racing-drones/, visited on 20.03.2022.

[38] MavLink. Ros node mavros. http://wiki.ros.org/mavros, https://github.com/mavlink/mavros, https://ardupilot.org/dev/docs/ros.html, all visited on 21.02.2022.

[39] Florian 'Floyd' Mueller and Matthew Muirhead. Jogging with a quad-

copter. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, page 2023–2032, New York, NY, USA, 2015. Association for Computing Machinery.

[40] Tobias Nägeli, Lukas Meier, Alexander Domahidi, Javier Alonso-Mora, and Otmar Hilliges. Real-time planning for automated multi-view drone cinematography. *ACM Trans. Graph.*, 36(4), jul 2017.

[41] Richard on Capolight Electronics Projects. The ideal shape for a multirotor 'ducted-fan' that can dramatically improve thrust and flight time. https://capolight.wordpress.com/2015/01/14/quadcopter-rotor-duct/, visited on 20.03.2022.

[42] Raspberry Pi. Raspberry pi 4 model b. https://www.raspberrypi.com/products/raspberry-pi-4-model-b/, visited on 20.02.2022.

[43] Vanessa Pirotta, David P. Hocking, Jason Iggleden, and Robert Harcourt. Drone observations of marine life and human - wildlife interactions off sydney, australia. *Drones*, 6(3), 2022.

[44] Maria Rubal. Sports to make you feel superhuman. https://www.delta.tudelft.nl/article/sports-make-you-feel-superhuman, visited on 05.02.2022.

[45] Judy Scott and C. Scott. Drone delivery models for healthcare. 01 2017.

[46] Superhuman Sports Society. #3 hado. https://superhuman-sports.org/sports/hado.php, visited on 26.03.2022.

[47] Superhuman Sports Society. #7 hado kart. https://superhuman-sports.org/sports/hadokart.php, visited on 26.03.2022.

[48] SSS. Superhuman sports society. https://superhuman-sports.org/, visited on 05.02.2022.

[49] T. Subha, Ranjana Raut, D. Kailash, and S. Abisha. *Drone Usage in Delivery of Vaccines in Indian Scenario*, pages 141–153. 03 2022.

[50] Rameesha Tariq, Maham Rahim, Nimra Aslam, Narmeen Bawany, and Ummay Faseeha. Dronaid : A smart human detection drone for rescue. In *2018 15th International Conference on Smart Cities: Improving Quality of Life Using ICT IoT (HONET-ICT)*, pages 33–37, 2018.

[51] TUM Phoenix Robotics Autonomous Flight Team. MS Windows NT indoor-drohne "golden snitch". https://www.epc.ed.tum.de/phoenix/autonomous-flight/indoor-drohne-golden-snitch/, visited on 05.02.2022.

[52] technik consulting. Setting the pid controller of a drone properly. https://www.technik-consulting.eu/en/optimizing/drone_PID-optimizing.html, visited on 20.03.2022.

[53] Terabee. Ros package for teraranger array solutions by terabee. https://github.com/Terabee/teraranger_array, visited on 22.03.2022.

[54] Terabee. Teraranger multiflex - modular and low-cost 8-sensor kit, 2m, 50hz, 20 grams. https://www.terabee.com/shop/lidar-tof-range-finders/teraranger-multiflex/, visited on 19.02.2022.

[55] Wings Over the Rockies Air & Space Museum. Drone soccer: The world's newest e-sport. https://www.youtube.com/watch?v=E6_ejnpyqBI, visited on 09.03.2022.

[56] Junya Tominaga, Kensaku Kawauchi, and Jun Rekimoto. Around me: A system with an escort robot providing a sports player's self-images. In *Proceedings of the 5th Augmented Human International Conference*, AH '14, New York, NY, USA, 2014. Association for Computing Machinery.

[57] Sergej Zwaan and Emilia Barakova. Boxing against drones: Drones in sports education. pages 607–612, 06 2016.