Guiding Quadrotor Landing with Pointing Gestures

Boris Gromov, Luca Gambardella, and Alessandro Giusti

Dalle Molle Institute for Artificial Intelligence (IDSIA), USI / SUPSI
Lugano, Switzerland
boris@idsia.ch

Abstract. We present a system which allows an operator to land a quadrotor on a precise spot in its proximity by only using pointing gestures; the system has very limited requirements in terms of robot capabilities, relies on an unobtrusive bracelet-like device worn by the operator, and depends on proven, field-ready technologies. During the interaction, the robot continuously provides feedback by controlling its position in real time: such feedback has a fundamental role in mitigating sensing inaccuracies and improving user experience. We report a user study where our approach compares well with a standard joystick-based controller in terms of intuitiveness (amount of training required), landing spot accuracy, and efficiency.

Keywords: pointing gestures, human-robot interaction, quadrotor landing

Videos, Datasets, and Code

Videos, datasets, and code to reproduce our results are available at:
http://people.idsia.ch/~gromov/hri-landing.

1 Introduction

Modern quadrotors can perform fully-autonomous missions in unstructured outdoor environments (e.g. for mapping, surveillance, etc); however, one delicate step in which human guidance, or at least supervision, is still necessary is landing, especially if this has to occur in an unstructured or populated environment. The landing spot should be sufficiently flat, dry, reasonably far from obstacles, and free of features such as long grass that would interfere with the rotors. Not surprisingly, many experienced pilots choose to land small quadrotors directly on their hand—a solution that is unsafe for the unexperienced operator, and unsuitable for larger quadrotors. In general, landing a quadrotor on an unpaved, unstructured surface requires a careful and critical choice of the landing spot by the operator. While approaches to automatically identify such potential landing spots by means of on-board sensing have been proposed in the literature [21, 9],

Fig. 1. One of the subjects pointing at the drone to select it (*left*), then guiding it (*center*) to land on a target (*right*).

it is realistic to assume that in many real-world scenarios the guidance of an operator in close proximity of the area will still be required, e.g. for avoiding puddles—which may look like perfectly-flat spots in a 3D reconstruction.

The standard approach for landing a quadrotor is by guiding it using a joystick; this is an efficient technique but requires the operator to be trained, and requires the use of both hands. We target scenarios in which the operator is not necessarily a trained pilot and may be in fact busy with other tasks. Such an example could be found in the future of search and rescue missions with mixed teams of humans and robots: a rescuer might need to land a quadrotor in its vicinity (for changing a battery, retrieving a carried object, etc.) without having a specific training; in this context, we assume that all potential operators wear an unobtrusive, networked bracelet-like device (e.g. a smartwatch) and that they can take control of a nearby drone to guide it to a safe landing spot.

Another realistic application is a drone delivery. Most part of such a mission can be performed autonomously without human intervention: the drone takes off and follows a set of way points to reach a house of the recipient, using for example Global Positioning System (GPS); however, the GPS localization accuracy in urban environments can be deteriorated and will not allow the drone to land autonomously in a safe manner. In this case, the recipient could guide the drone to a safe landing spot with a poining gesture.

We focus on this well-defined task and aim at providing a control modality with the following characteristics.

- *Practical/pragmatic*: has no strong requirements on the drone or infrastructure capabilities and can be robustly applied to real-world systems in many realistic conditions (outdoors, indoors).
- *Intuitive*: operators need minimal training to use it.
- *Efficient*: an operator can land a drone in a short time.

The interaction begins when the drone approaches a designated position after completing an autonomous part of a mission. To initiate the landing procedure an operator, e.g. a rescuer or a parcel recipient, points at the drone to select it. In turn, the drone provides a visual feedback to confirm that the control has been transferred to the operator, e.g. by performing a predefined motion primitive or by blinking with its on-board lights. From that moment, the drone follows and hovers over the location on the ground pointed by the operator. Once the

operator is determined to land the drone, they simply keep the arm still for a predefined time. A countdown timer starts and the system sends periodic feedback to the user to notify them that the drone is about to land. In case the operator decides to adjust the landing spot, they simply point at another location, the countdown timer cancels and the landing procedure starts over again. This interaction sequence is shown in brief in Figure 1, please refer to the linked video for details.

In order to prove the viability of this control approach, we implemented it and experimentally compared its performance with joystick-based control.

In the following sections we review the related work (Section 2), define the major abstract functionalities required to realize the given interface and task, and discuss implementation options (Section 3). We describe our implementation of these functionalities in Section 4. To compare our approach with the classic joystick-based control we set up an experiment, which is described in detail in Section 5, the results are then analyzed in Section 6. Finally, in Section 7 we draw the conclusions and describe the future work.

2 Related Work

Since pointing gestures are such a compelling solution to many human-computer and human-robot interaction problems, significant research efforts have been devoted to this topic. To the best of our knowledge, however, this work is the first to approach the issue of landing a drone by using pointing gestures.

There are many works that use iconic gestures [22] to control drones [17, 20]. These can use hands, arms, or full-body postures to give discrete commands to the drones, such as "go up", "go down", "turn left", "take off", etc. Although these gestures may be represented by a pointing hand or arm, the exact direction of this pointing is not important. On the contrary, we are interested in pointing gestures that indicate precise directions and locations with respect to the user. Therefore, below we only review the research related to this particular type of pointing gestures and omit the interfaces based on iconic gestures.

Using pointing gestures as an input interface dates back to 1980s, when Bolt presented his now famous work "Put-that-there" [2]. A multi-modal human-computer interaction interface developed in that work was used to manipulate virtual objects on a screen. The input interface consisted of a commercial speech recognition system and a pose sensing device. The pointing device included a stationary transmitter of a nutating magnetic field and a small wired sensor placed on the user's wrist. Altogether the system allowed to manipulate objects using simple voice queries like "Put that there", where "that" would be supported by one pointing gesture and "there" by another. In this case, Bolt argues, the user even does not have to know what the object is or how it is called.

In HRI literature, pointing gestures are often used for pick-and-place tasks [4, 6, 5], labeling and/or querying information about objects or locations [4], selecting a robot within a group [16, 19], and providing navigational goals [25, 1, 26, 14, 12].

One important issue to be solved in natural human-robot interaction that involves pointing is a perception of the user's gestures. This can be a responsibility of a robot, i.e. the recipient of the message, as well as of a group of cooperatively-sensing robots [19]; of the environment [27]; or, as in our case, of a device worn by the user [23, 26, 12]. The first approach is the most popular in HRI research. On one hand, it is natural because it mimics what humans do when communicating with each other (the recipient of the message is the one which perceives it). On the other hand, it presents important challenges to solve the perception problem, and requires the robot to consistently monitor the user. Relying on sensors placed in the environment relaxes the requirements on the robots, but limits the applicability of the system to properly instrumented areas; in both cases, the positions being pointed at need to be inferred by external observation, which is typically performed with cameras or RGB-D sensors.

2.1 Providing navigational goals

We now focus our review on pointing gestures for robot guidance.

Van den Bergh [25] used pointed directions to help a ground robot to explore its environment. The robot continuously looks for a human in its vicinity and once detected begins the interaction. Using an RGB-D sensor (Microsoft Kinect) the system detects the human's hand and wrist. A vector connecting the center of the hand and the wrist is then projected on the ground, giving a principal exploration direction. Finally, the next exploration goal is automatically selected from a set of possible goals with respect to an instantaneous occupancy grid acquired by the robot. The authors also suggest an alternative method to estimate pointing directions, namely a line connecting the eyes and the fingertip, however they do not elaborate on this approach.

Similarly to the previous work, Abidi et al. [1] use a Kinect sensor to extract pointed directions. Navigation goals are continuously sent to the ground robot, which reactively plans its motion and thus allows the user to correct her input on the fly. The main drawback, however, is that the robot has to "keep an eye" on the user in order to reach the final target. To estimate pointed locations authors suggest two approaches: (1) a vector originating from the elbow and passing the hand/finger, and (2) a vector originating from the eyes and also passing the hand/finger. The approaches were compared in a user study, but the only reported result is a subjective satisfaction level of the participants. The majority (62%) preferred the second approach.

Jevtic et al. [14] experimentally compared several interaction modalities in the user study of 24 participants. A ground robot equipped with a Kinect and other sensors was used. The study compares three interaction modalities: direct physical interaction (DPI), person following, and pointing control in area- and waypoint-guidance tasks. The DPI modality requires the user to push the robot by hands, the torques generated at motors are measured via electrical current and then are fed to a friction-compensation controller that drives the robot in the appropriate direction. The person following modality makes the robot to follow the user at a safe distance, the user can stop the robot at any time by

raising their left hand above the left elbow and thus can control the robot's precise location. The pointing modality allows the user to command the robot's position with a pointing gesture, where the target location is calculated from the intersection of the ground plane with a line passing through the right elbow and the right hand of the user. The authors measured task completion times, accuracy, and workload (with NASA-TLX questionnaire). Reported results show that the DPI modality is systematically better than the other modalities for all the metrics, while the pointing control shows the worst results.

Such a low performance of the pointing interface used in the study by Jevtic et al. [14] can be explained by a lack of appropriate feedback and a time-sparse nature of the implemented gesture control: the user issues a single command to drive the robot to a goal and see where the system "thinks" they were pointing at only when the robot reaches the target, therefore, the user is unable to efficiently correct the robot's position. These problems are further aggravated by an inherently limited precision of a chosen pointing model (elbow-hand). As reported by many other works (see [1, 18, 6]), including those from the psychology research (see [24, 13])—a more appropriate model would be a line that passes through the head and the fingertip. Also note that contrary to the implemented pointing control, the DPI and person following modalities work in the tracking mode, that provides immediate feedback and allows the user to correct robot's position in the real time. As will be seen later, in our work we mitigate aforementioned flaws by making the robot to instantaneously follow to pointed locations, and thus providing a real time feedback to the user. In our recent work [10] we systematically compared users performance in a pointing task with and without the visual feedback: we have shown that the lack of visual feedback results in significant errors.

2.2 Wearable sensors

Wearable sensors are an alternative approach to the problem of perceiving pointing gestures. Sugiyama et al. [23] developed a wearable visuo-inertial interface for on-site robot teaching that uses a combination of monocular camera and inertial measurement unit (IMU) to capture hand gestures in the egocentric view of the user. The camera is also used for a monocular simultaneous localization and mapping (SLAM), which allows to localize the user with respect to a common with the robot coordinate frame.

Wolf et al. [26] suggest a gesture-based interface for a robot control, that is based on a device they call BioSleeve—a wearable device placed on the user's forearm and comprised of a set of dry-contact surface electro-myography sensors (EMGs) and an IMU. Optionally, the authors suggest to strap an additional IMU sensor on the upper arm to be able to perform a model-based arm pose reconstruction for pointing gestures. However, no information is given on how a user would localize themselves with respect to the robot in order to control its position.

2.3 Pointing direction estimation

Regardless on the specific approach adopted for sensing, assuming a perfect knowledge of the user's posture, one has to solve the problem of interpreting such posture to map it to the point in the environment that the user wants to indicate. This problem has been extensively studied in the psychology research [24, 13] which suggests two main models: a) the *forearm* model assumes that the point lies along the 3D line defined by the axis of the forearm of the pointing arm; b) the *head-finger* model assumes that the point lies along the line connecting the dominant eye and the tip of the finger. The choice of a particular model in robotic applications mainly depends on the technology available for sensing the user's posture and on the task.

3 Model

In order to realize the task we have defined in the introduction, it is necessary to address the following problems: a) estimation of the pointed directions and locations with respect to the human, b) identification and localization of a robot being pointed at, and c) detection of discrete triggering events that would dispatch commands to the robot.

Pointed direction The pointed direction is recovered as a ray in 3D space, expressed in a human-centered reference frame. The frame is fixed while the interaction occurs and has its origin at user's feet; its xy-plane is aligned with the world's horizon; the remaining degree of freedom (a rotation around the vertical axis) is a free parameter. Without loss of generality, we can assume the x-axis is the heading of the first pointing gesture, i.e. the one used to select the robot.

A forearm-mounted IMU is a viable option to meet this requirement as long as it can return accurate relative orientation data without significant drift for the duration of the interaction, and can reliably estimate the vertical direction. A more sophisticated approach may use additional sensors, e.g. multiple IMUs [26, 12], to model more accurately the arm kinematics. Since the IMUs provide only a 3D-rotation, one will also need to acquire the position of the user's shoulder with respect to the human frame. It can be measured directly or estimated using a simple calibration procedure.

Robot identification and pose reconstruction Since pointed direction is expressed in human's reference frame, it is necessary to find a coordinate transformation between the human and a robot. We require that while a robot is being pointed at, the system can detect it, identify it, and recover its 6D-pose (3D-position + 3D-orientation) with respect to the human-centered frame.

In practice, this can be achieved in numerous ways: with a camera pointed in the same direction as the forearm [12], which can detect the robot's presence

and identify its pose using pattern recognition techniques, e.g. relying on robot-mounted visual fiducial markers [8], or detecting active LEDs [7].

Recently we described another efficient method to co-localize a user and a robot [11]. The method relies on synchronized motion of user's arm and the robot: the user points and keeps following a moving robot they want to interact with for a few seconds; the system collects synchronized pairs of pointing rays expressed in user's frame and robot's positions expressed in its odometry frame, the algorithm then finds the coordinate transformation between the human and the robot that fits the captured movements the best.

Triggering We assume that the system implements a mechanism to trigger the first and second pointing events (at robot and at target) and thus advance the interaction.

A trivial approach is to use a push button on the wearable/handheld device, which however prevents hands-free operation. Other realistic triggering mechanisms include gestures, automatic detection of the start of a pointing gesture [18], fixed time delays [5], or speech [12].

4 Implementation

We implemented a pointing-based interface that consists of two Myo armbands, respectively placed on the upper arm and the forearm. We use a single sensor for the arm pose estimation, however we use both for the gesture detection which is described later.

Myo is an integrated wireless wearable sensor from Thalmic Labs[1] comprised of 9-DoF IMU, eight surface electromyography sensors (EMG) and a processing unit. The device internally fuses the IMU data into an accurate absolute 3D-orientation (roll, pitch, and yaw angles) which we use for the arm pose estimation. The inertial data is transmitted to the host PC over the Bluetooth LE link at 50 Hz rate.

Pointed direction To estimate pointed directions we employ a *head-finger* model—a popular choice in robotics [6, 5, 18, 15]—which requires the knowledge of the arm and head positions. However, we simplify this model and assume that: a) the head position is fixed with respect to the body; b) the user always points with the straight arm, and c) the shoulder length (distance from the neck to the shoulder joint) is zero.

Although these simplifications lead to higher lateral and radial static errors, a live update of the drone's position allows to efficiently mitigate them.

In this work we consider the arm as a single link with a 3-DoF ball joint (shoulder) connected to a fixed vertical link (torso). We acquire the orientation of the shoulder with a help of a single Myo armband placed on the forearm.

[1] Myo has been discontinued as of Oct 12, 2018.

Once we know the pointing direction with respect to the human's shoulder we can simply find the pointed location as an intersection of a line and the ground plane.

Robot identification and pose reconstruction Since there is only one robot to control, there is no need for robot identification.

In order to determine the robot's location with respect to the operator, we assume that the robot is flying at a known altitude over a flat floor; under this assumption, the distance to the robot can be estimated from the pointing ray alone, by intersecting it with the horizontal plane on which the robot is flying.

The only remaining parameter is the relative heading between the operator and the drone. Most of the drones are equipped with an IMU for stabilization purposes, and usually include a magnetometer—the sensor that estimates the heading with respect to the Magnetic North, an absolute reference frame. Therefore, the rotation between the frames is defined by the difference between the human's and robot's absolute headings.

Triggering Using data from two wireless IMU sensors worn on the arm and forearm, we follow a detection-by-classification paradigm and use a 1D convolutional neural network as a binary classifier. Given the data acquired in the last few seconds, the network predicts whether a pointing gesture occurred in this interval. The network has been trained using data acquired from multiple users, who were prompted by the system to perform the gesture at specific times [3].

Therefore, the action is triggered immediately once the user performs the pointing gesture.

5 Experimental Setup

To confirm the viability of the proposed interface in a guided landing task, we set up an experiment where the users are required to land a quadrotor (Parrot Bebop 2) at a given location using two different interfaces: pointing gestures and a regular joystick.

The experimental environment represents a flying arena with four predefined targets. The targets are placed at the corners of a square with an edge of 3.6 m and numbered in clockwise order. The sequence of the targets is predefined as 1–2–4–3–1, i.e. edge segments alternate with diagonal ones. The subjects were asked to stay in the middle of arena, however they were allowed to step aside to avoid collisions with the drone.

The arena is equipped with the Optitrack motion capture system that provides precise information of the drone's position. This information is used, both, to control the safety margins and to implement autonomous flights. Note, that in general the robot is localized in an arbitrary frame, e.g. in its odometry frame or, as in our case, in the motion capture frame; however, the location of the operator with respect to robot's frame is not known until the "Robot identification and pose reconstruction" step has taken place.

The drone closed-loop controller is built around `bebop_autonomy`[2] ROS-package and accepts velocity and 6D-pose commands. While the joystick interface generates velocity commands, the pointing interface supplies the pose commands.

5.1 Subjects

Five people between 25 and 36 years old have volunteered to participate in the experiment. Majority have reported either no experience in piloting RC-vehicles or a little experience ("tried it a few times").

We conducted two sessions per person: with the joystick and with the pointing-based interface, each consisting of three runs. Each run starts and ends at target 1 and therefore provides four segments. This totals to 12 segments per person or 120 target-to-target segments for all the participants for both interfaces. Three subjects started with the pointing interface and the rest with the joystick.

Prior to each experimental session, individually for each subject, we conducted a training session with the same interface that they were given later. Each training session consisted of two runs.

Two training sessions plus two experimental sessions took approximately 40 minutes per person, including all the explanations, service times (e.g. replacing the drone's battery), etc.

5.2 Experimental sequence

At the beginning of each session the drone is placed at target 1. Once the supervisor starts the session the drone takes off automatically. Once it is airborne and stable, it aligns itself with the first target of the segment and turns its back to the user, such that the user's controls, both, for the joystick and for the gestures, are aligned, meaning, e.g., that pushing joystick forward would drive the drone away from the user, in the direction they look to. This way we ensure that the landing errors are not accumulated over the course of experiment and that all the subjects start in equal conditions, both, when controlling the drone with pointing gestures and with a joystick.

From this moment the drone is ready to interact with the user. Once the user lands the drone, it will perform the automatic take off procedure with a delay of 5 seconds from the moment it has been landed. This procedure repeats until the list of targets is exhausted.

Since the interaction pattern with the joystick and the gesture-based interface are slightly different we report them separately.

Pointing-based interface The user points at the floor beneath the drone, i.e. at the crosshair of the target, to select it[3]. The Myo on the user's arm vibrates

[2] http://wiki.ros.org/bebop_autonomy

[3] Although the drone can be selected in the air, this brings additional error to the relative localization and may deteriorate user experience.

and the drone 'jumps' to signify that it is now being controlled by the user. At the same time, the control station gives a voice feedback through a loudspeaker, telling the user the next target they should bring the drone to. Immediately after that the drone starts to continuously track a newly given location. Once the user is ready to land the drone, they have to maintain the pointed location for approximately half a second. The system starts to count down and makes the upper arm Myo to vibrate every second. The user has about $3s$ to change their mind. To adjust the landing position they just have to start moving the arm away and the countdown will be canceled.

Joystick The behavior of the system in this mode is similar, however the drone is selected automatically and performs the same 'jump' motion as in pointing-based interaction mode, meanwhile the control station gives a voice feedback with the next target number. The user then moves the drone to the required target and presses the button on the joypad to land the drone.

5.3 Performance Metrics

We define a set of performance metrics to compare the performance of two interfaces:

- *Landing error.* Euclidean distance between the requested landing target P_i and the actual position p_a the drone have landed to: $\varepsilon = |p_a - P_i|$.
- *Time to target.* Time that passed from the moment t_0 the drone has moved $20cm$ away from its starting pose till the moment t_1 the user gave command to land it: $\tau = t_1 - t_0$.
- *Trajectory length.* Line integral of the trajectory curve from the start position P_{i-1} to the actual landing position p_a: $\rho = \sum_{k=1}^{N} \|p_{k+1} - p_k\|$, where $p_k \in \{p_1, ...p_N\}$ is a set of all the acquired positions of the drone between P_{i-1} and p_a.

We collect the data with a standard ROS tool `rosbag` and analyze it offline.

6 Results

Landing Error Figure 2 reports the landing error metric. We observe no statistically significant difference between the results with the two interfaces; we separately report the error (left pair) and its decomposition in a radial (central pair) and tangential (right pair) component with respect to the user's position. It's interesting to note how the radial component dominates the error in both interfaces, which is expected since the radial component corresponds to the depth direction from the user's perspective: along this direction, the quadrotor's misalignment with respect to the target is much more difficult to assess visually; the landing error is dominated by a perception (rather than control) issue.

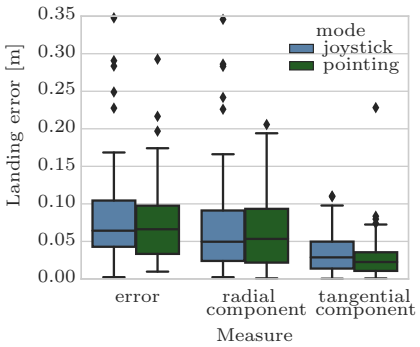

Fig. 2. Statistical analysis of the landing error metric ($N = 60$ for each interface).

Time to Target Figure 3 (*left*) reports the time to target metric, separately for each of the four segments. We observe that the pointing interface yields a better average performance; the difference is statistically significant under Student's t-test ($p < 0.01$) for 3 of the 4 segments and for the mean over all segments.

Trajectory Length Figure 3 (*right*) reports the trajectory length metric, separately for each of the four segments. We observe that the pointing interface consistently yields shorter trajectories than the joystick interface; the difference is statistically significant under Student's t-test ($p < 0.01$) for all four segments. On average over all segments, the joystick interface yields a 66% longer trajectory than the straight distance between the targets; the pointing interface yields a 33% longer trajectory.

One can also observe this phenomenon in Figure 4: the trajectories flown with the pointing interface are smoother and more direct and tend to converge faster to the target (Figure 5).

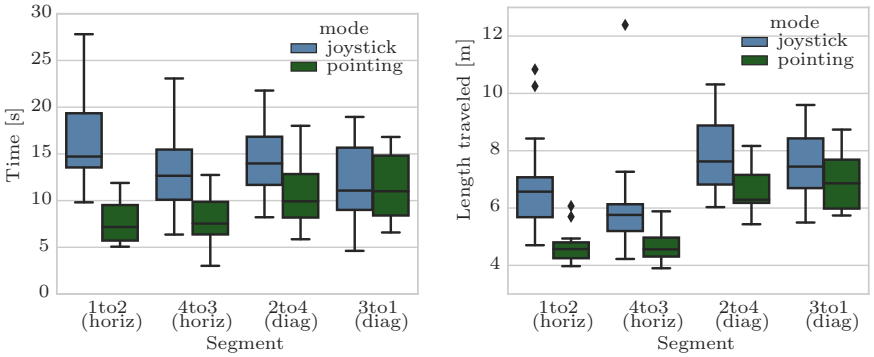

Fig. 3. Statistical analysis of the time to target (*left*) and trajectory length metrics (*right*) ($N = 15$ for each interface and segment).

7 Conclusions

We proposed a novel human-robot interface for landing a quadrotor based on pointing gestures detected by means of unobtrusive wearable sensors. The interface has minimal requirements in terms of robot capabilities, and extensively takes advantage from real-time feedback. In a preliminary user study, it compares favorably with a traditional joystick-based interface in terms of efficiency and intuitiveness.

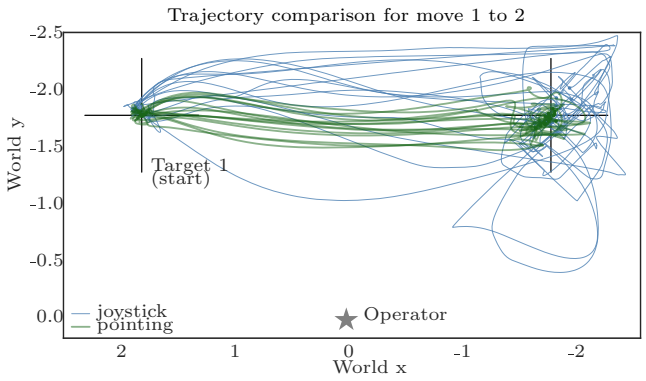

Fig. 4. Comparison of all trajectories flown from target 1 (left) to target 2 (right) for joystick (blue, $N = 15$) and pointing (green, $N = 15$) interfaces.

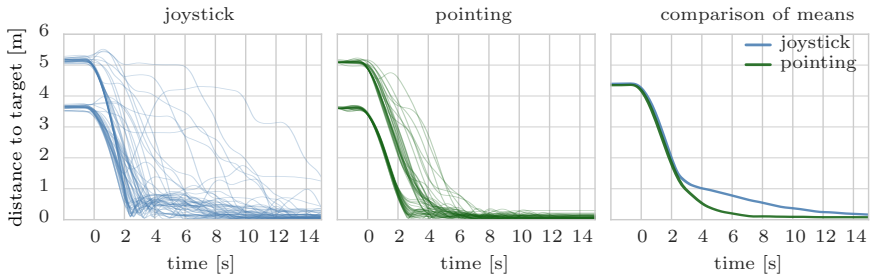

Fig. 5. Evolution in time of the distance to the target, for each trajectory flown: (*left*) joystick interface ($N = 60$), (*center*) pointing interface ($N = 60$), (*right*) average over all trajectories for each interface.

Acknowledgments

This work was partially supported by the Swiss National Science Foundation (SNSF) through the National Centre of Competence in Research (NCCR) Robotics.

Bibliography

[1] Abidi, S., Williams, M., Johnston, B.: Human pointing as a robot directive. ACM/IEEE International Conference on Human-Robot Interaction pp. 67–68 (2013)

[2] Bolt, R.A.: "Put-that-there": Voice and Gesture at the Graphics Interface. Proceedings of the 7th annual conference on Computer graphics and interactive techniques - SIGGRAPH '80 pp. 262–270 (1980)

[3] Broggini, D., Gromov, B., Gambardella, L.M., Giusti, A.: Learning to detect pointing gestures from wearable IMUs. In: Proceedings of Thirty-Second AAAI Conference on Artificial Intelligence, 2018, USA. AAAI Press (2018)

[4] Brooks, A.G., Breazeal, C.: Working with Robots and Objects: Revisiting Deictic Reference for Achieving Spatial Common Ground. Gesture pp. 297–304 (2006)

[5] Cosgun, A., Trevor, A.J.B., Christensen, H.I.: Did you Mean this Object?: Detecting Ambiguity in Pointing Gesture Targets. In: HRI'15 Towards a Framework for Joint Action Workshop (2015)

[6] Droeschel, D., Stückler, J., Behnke, S.: Learning to interpret pointing gestures with a time-of-flight camera. Proceedings of the 6th international conference on Human-robot interaction - HRI '11 pp. 481–488 (2011)

[7] Faessler, M., Mueggler, E., Schwabe, K., Scaramuzza, D.: A monocular pose estimation system based on infrared leds. In: Robotics and Automation (ICRA), 2014 IEEE International Conference on, pp. 907–913. IEEE (2014)

[8] Fiala, M.: Designing highly reliable fiducial markers. IEEE Transactions on Pattern Analysis and Machine Intelligence **32**(7), 1317–1324 (2010)

[9] Forster, C., Faessler, M., Fontana, F., Werlberger, M., Scaramuzza, D.: Continuous on-board monocular-vision-based elevation mapping applied to autonomous landing of micro aerial vehicles. In: 2015 IEEE International Conference on Robotics and Automation (ICRA), pp. 111–118 (2015)

[10] Gromov, B., Abbate, G., Gambardella, L., Giusti, A.: Proximity human-robot interaction using pointing gestures and a wrist-mounted IMU. In: 2019 IEEE International Conference on Robotics and Automation (ICRA), pp. 8084–8091 (2019)

[11] Gromov, B., Gambardella, L., Giusti, A.: Robot identification and localization with pointing gestures. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3921–3928 (2018)

[12] Gromov, B., Gambardella, L.M., Di Caro, G.A.: Wearable multi-modal interface for human multi-robot interaction. 2016 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR) pp. 240–245 (2016)

[13] Herbort, O., Kunde, W.: Spatial (mis-) interpretation of pointing gestures to distal spatial referents. Journal of Experimental Psychology: Human Perception and Performance **42**(1), 78–89 (2016)

[14] Jevtić, A., Doisy, G., Parmet, Y., Edan, Y.: Comparison of Interaction Modalities for Mobile Indoor Robot Guidance: Direct Physical Interaction,

Person Following, and Pointing Control. IEEE Transactions on Human-Machine Systems **45**(6), 653–663 (2015)
[15] Mayer, S., Wolf, K., Schneegass, S., Henze, N.: Modeling Distant Pointing for Compensating Systematic Displacements. In: Proceedings of the ACM CHI'15 Conference on Human Factors in Computing Systems, vol. 1, pp. 4165–4168 (2015)
[16] Nagi, J., Giusti, A., Gambardella, L.M., Di Caro, G.A.: Human-swarm interaction using spatial gestures. In: IEEE International Conference on Intelligent Robots and Systems, pp. 3834–3841 (2014)
[17] Ng, W.S., Sharlin, E.: Collocated interaction with flying robots. Proceedings - IEEE International Workshop on Robot and Human Interactive Communication pp. 143–149 (2011)
[18] Nickel, K., Stiefelhagen, R.: Visual recognition of pointing gestures for human-robot interaction. Image and Vision Computing **25**(12), 1875–1884 (2007)
[19] Pourmehr, S., Monajjemi, V., Wawerla, J., Vaughan, R., Mori, G.: A robust integrated system for selecting and commanding multiple mobile robots. Proceedings - IEEE International Conference on Robotics and Automation pp. 2874–2879 (2013)
[20] Sanna, A., Lamberti, F., Paravati, G., Manuri, F.: A Kinect-based natural interface for quadrotor control. Entertainment Computing **4**(3), 179–186 (2013)
[21] Scherer, S., Chamberlain, L., Singh, S.: First results in autonomous landing and obstacle avoidance by a full-scale helicopter. Proceedings - IEEE International Conference on Robotics and Automation pp. 951–956 (2012)
[22] Suarez, J., Murphy, R.R.: Hand gesture recognition with depth images: A review. Ro-Man, 2012 Ieee pp. 411–417 (2012)
[23] Sugiyama, J., Miura, J.: A wearable visuo-inertial interface for humanoid robot control. In: ACM/IEEE International Conference on Human-Robot Interaction, pp. 235–236. IEEE (2013)
[24] Taylor, J.L., McCloskey, D.: Pointing. Behavioural Brain Research **29**(1-2), 1–5 (1988)
[25] Van den Bergh, M., Carton, D., De Nijs, R., Mitsou, N., Landsiedel, C., Kuehnlenz, K., Wollherr, D., Van Gool, L., Buss, M.: Real-time 3D hand gesture interaction with a robot for understanding directions from humans. Proceedings - IEEE International Workshop on Robot and Human Interactive Communication pp. 357–362 (2011)
[26] Wolf, M.T., Assad, C., Vernacchia, M.T., Fromm, J., Jethani, H.L.: Gesture-based robot control with variable autonomy from the JPL BioSleeve. Proceedings - IEEE International Conference on Robotics and Automation pp. 1160–1165 (2013)
[27] Zivkovic, Z., Kliger, V., Kleihorst, R., Danilin, A., Schueler, B., Arturi, G., Chang, C.C., Aghajan, H.: Toward low latency gesture control using smart camera network. In: Computer Vision and Pattern Recognition Workshops, 2008. CVPRW'08. IEEE Computer Society Conference on, pp. 1–8. IEEE (2008)