Intuitive 3D Control of a Quadrotor in User Proximity with Pointing Gestures

Boris Gromov, Jérôme Guzzi, Luca M. Gambardella, Alessandro Giusti

***Abstract*— We present an approach for controlling the position of a quadrotor in 3D space using pointing gestures; the task is difficult because it is in general ambiguous to infer where, along the pointing ray, the robot should go. We propose and validate a pragmatic solution based on a push button acting as a simple additional input device which switches between different virtual workspace surfaces. Results of a study involving ten subjects show that the approach performs well on a challenging 3D piloting task, where it compares favorably with joystick control.**

Videos, Datasets, and Code

Video, datasets, and code to reproduce our results are available at: http://people.idsia.ch/~gromov/3d-pointing

I. Introduction

Pointing gestures are an attractive modality for human-robot interaction; it is pervasive in human-human communication and thus natural and intuitive to use. In our everyday lives, we point to give directions, refer to places, objects, and even events that happened in the past [1]. In human-robot interaction, pointing is a popular choice for tasks that require explicit and precise position information: for example, to tell an assistant robot to pick up a tool or to move to a location.

We consider pointing as an input interface for controlling robots within proximity and within visual contact of an operator. The operator controls the mobile robot by pointing at a desired target position p_t in 3D space for the robot to reach; the robot moves there, and keeps tracking the updated position of p_t when the operator points at another location. This allows the operator to finely control the robot in a continuous fashion, and drive it along complex trajectories.

This control modality is attractive because it features no indirection between human commands and robot actions. The operator acts directly in their local reference frame without the need to account for a possible coordinate frame mismatch between the input interface and the robot. Once the operator has identified a target for the robot, the cognitive effort required to point at it is minimal.

In our previous work [2], we have shown that pointing can be efficiently used to guide flying robots on complex 2D trajectories constrained to a horizontal plane at a predetermined height over a horizontal floor. This work extends the approach and proposes an efficient way to control flying robots also in 3D. Most operators who tested our previous prototypes expressed the wish for such a feature, but designing it faces a fundamental challenge: the act of pointing by itself does not uniquely identify the desired target position p_t. In fact, a given pointing stance defines a pointing ray r in 3D space, originating at the operator's body and extending to infinity along a given 3D direction: the desired target position p_t might lie anywhere on r.

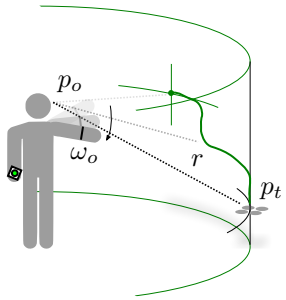

Fig. 1. The operator controls a quadrotor in 3D by pointing. A pointing ray r originating at p_o intersects a cylindrical workspace surface at p_t and fully defines the target position for the quadrotor, which tracks it in real time. In this paper, we investigate how, by switching workspace surfaces using a push button, an operator can freely control the robot's position in 3D.

In certain cases we can use additional assumptions to determine the target point p_t on the pointing ray: 1) If the robot's motion is constrained to a plane (e.g. a ground robot on flat ground, or a flying robot moving at a fixed height) one can identify p_t as the intersection between r and such plane, as in our previous work [2]; 2) If the target p_t lies very close to a solid surface of the environment (e.g. a ground wheeled or legged robot that travels on a generic terrain, including indoor environments with stairs; or a flying inspection robot that follows a complex 3D structure), p_t can be identified by intersecting r with a known model of the world's surfaces.

In this paper we consider a control of a quadrotor that *freely* moves in 3D space, where neither of these constraints apply; it is therefore impossible to unambiguously determine p_t from r; we further assume that the operator controls the robot in a continuous fashion accounting for their perception of the robot's position. Even with this assumption, the problem is underconstrained: consider the case in which the operator is pointing at the robot while it flies few centimeters from the ground and one meter in front of them. The operator then adjusts their pointing stance by slightly increasing the elevation of their arm. Does this mean that the robot should move farther from the user while staying at the same height, or that it should move up?

(*) Boris, Jérôme, Luca, and Alessandro are with the Dalle Molle Institute for Artificial Intelligence (IDSIA USI-SUPSI), Lugano, Switzerland. Email: {boris,jerome,luca,alessandrog}@idsia.ch.

After reviewing related work (Section II), we describe our **main contribution** (Section III): a pragmatic solution to the above problem, which relies on a well-defined *user-centric* set of *virtual* workspace surfaces that do not depend on the environment; the user switches between these surfaces with an additional control input; in practice, such input could be a push button held in either the pointing or free hand (see Figure 1), or a detector of specific gestures or simple voice commands. Section IV describes our implementation that is experimentally assessed in Section V, and Section VI reports experimental results from our user study.

II. Related Work

A large number of works on gesture-based drone interaction are based on *iconic* gestures [3, 4, 5]. These can use hands, arms, or full-body postures to give discrete commands to a drone, 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, we only review the research related to this type of pointing gestures and omit interfaces based on iconic gestures.

Pointing gestures is a skill that humans develop since an early age [6]. They are used in everyday lives to efficiently communicate locations and directions in the surrounding 3D space to other people. It also has been shown that people naturally choose to use pointing gestures when they need to command a drone to move to a precise location [7]. For these reasons pointing gestures are a popular choice in robotics, where they are typically used for tasks such as pick-and-place [8, 9, 10], object and area labeling [11], teaching by demonstration [12], point-to-goal [13, 14], selection of a robot within a group [15, 16], and assessment of joint attention [17].

The problem of using pointing gestures consists of two parts: perception of the gesture itself and estimation of the pointed location. The perception of gestures can be performed by a robot or by a group of robots [18, 16], by instrumented environment [19], or, as in our case, by a device worn by the user [12, 13, 15]. The first approach is the most popular in human-robot interaction (HRI) research, however it requires solving a challenging perception problem because the robot has to continuously sense 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.

Regardless of the specific approach adopted for sensing, assuming a perfect knowledge of the user's posture, one has to solve the problem of interpreting such a posture and mapping it to the point in the environment that the user wants to indicate; this is typically solved in two steps: first, identify a direction (i.e. a ray in 3D space); then, relate such ray with the environment to get a point or object. The first step is typically solved by defining a ray that originates at head ([20, 21, 22, 23] or at arm ([22, 8, 24, 25]) and passes through another point located on the arm, for example, tip of the index finger. Different configurations of these two points define a number of pointing models, in particular in robotics [8, 10, 22, 24]: *head-finger*, *upper arm*, and *forearm* models. In this work, we employ the head-finger model.

Once the pointing direction is found one needs to identify the pointed location or object. This is typically done by intersection of the pointing ray with the environment. For example, with large public displays [26, 27, 28] or with the model of the environment acquired with depth sensors [8, 9, 10].

Our goal, however, is to define a target 3D position in free space. An interesting approach to this problem is to use motion scaling within pick-and-place paradigm [29]: the user "picks" a drone with the pinch gesture as if it is within arm's reach and moves the hand to a new location to "place" the drone there; the 3D offset of the hand defines a displacement vector that is applied to the drone but is scaled proportionally to the user–drone distance. A similar image plane manipulation technique was earlier also proposed for object manipulation in virtual environments [30]. This approach has two drawbacks: 1) because the hand's and drone's displacement vectors are parallel it might be difficult for a user to visualize where the drone will end up; 2) inherently limited range of motion of the hand limits the range of motion of the drone.

We approach the problem of defining 3D positions in free space by introducing virtual workspace shapes that constrain the motion of the robot on a given surface, e.g. plane, cylinder, or sphere. As compared to image plane manipulation techniques, our approach allows one to easily predict the target position of the robot and, technically, allows the user to position a robot in entire 3D space.

III. Model

A. Target point

In this work, we assume that the robot and the operator are localized in a common reference frame. In practice, we can estimate the transformation between their relative frames with collaborative visual-inertial SLAM methods [31] or a technique that compares the motion of a robot and an arm that points at it [32].

The operator indicates a target point p_t by pointing. We assume that p_t lies on a *pointing ray* r. To identify r, we adopt a simplified version of the *head-finger* model (eye-finger ray cast method by Mayer et al. [21]) which defines r as the half-line originating at the operator's dominant eye at point p_o and passing through the tip of the pointing finger, located at the end of a straight arm with orientation ω_o, which is measured by a wrist-mounted IMU.

With the further assumption that eye and shoulder are vertically aligned (and that shoulder height, shoulder-finger length, shoulder-eye distance are known), we can reconstruct $r = r(\omega_o; p_o)$. When the origin p_o of the ray is fixed, the

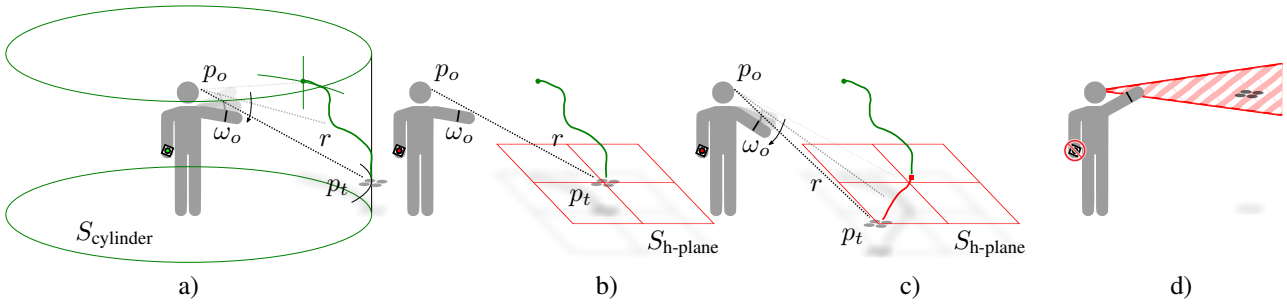

Fig. 2. Interaction using workspace shapes: a) guiding the drone in the primary workspace S_{cylinder}; b) switching to the secondary shape by pressing a button; c) guiding the drone in the secondary workspace $S_{\text{h-plane}}$; d) while the drone flies close to eye height, the user is forbidden to switch the workspace to $S_{\text{h-plane}}$: in this case the pointing ray is almost parallel to the virtual surface, which prevents accurate control of the drone's distance.

operator has two degrees of freedom (the arm orientation) to move r and point to a different target.

We define p_t as the intersection of the pointing ray r with a *workspace surface* S: $p_t = r \cap S$, i.e., the operator, by moving their arm, moves a well-defined 3D target point on a two dimensional surface.

We have previously shown [2] that intersecting the pointing ray with a *horizontal plane* is effective to define target positions for a ground robot or a quadrotor constrained to fly at a fixed height. In this work, we let the operator switch between *different* workspace surfaces (see Figure 2) to enable quadrotor control in the entire 3D space.

B. Workspace shapes

We considered several options for workspace shapes: vertical-axis cylinder, horizontal plane, sphere, and vertical plane. We now describe the first two: their combination allows the operator to reach any position in 3D space efficiently and in an intuitive way. The supplementary appendix motivates this choice in detail and discusses drawbacks of other options.

Each shape is defined as a one parameter family of surfaces: when the user switches to the shape, the free parameter is set in such a way that the surface passes through the current position p_{robot} of the robot.

a) Cylinder $S_{cylinder}$: with a vertical axis passing through the user's head (p_o); the cylinder radius is the free parameter. This option allows the operator to control the robot's vertical position without limitations, never affecting the horizontal distance to the operator.

b) Horizontal plane $S_{h\text{-}plane}$: with the distance from the ground plane as the free parameter. This workspace serves a similar role to the ground plane that humans have a life-long experience with: it is a natural choice for indicating locations and an intuitive tool to control the robot's position on that plane, but does not allow height control.

To achieve intuitive interaction, an operator should always have a clear idea of the workspace the robot is operating in; if the workspace shape is known, *the robot position itself* uniquely defines the workspace surface S. For example, if the workspace shape is $S_{\text{h-plane}}$, the user can expect that the robot will keep its current vertical position; if the workspace shape is S_{cylinder}, one can easily visualize the user-centered cylinder passing through the robot. In turn, if S is known, the user can always predict p_t given r.

IV. Implementation

A. Gesture sensing

We implemented the system using an inexpensive wearable IMU (Mbientlab MetaWearR+ [33]) that has a form-factor of a wrist smartwatch (Figure 3, right). The device is equipped with a three degrees of freedom (3-DoF) accelerometer, 3-DoF gyroscope, and 3-DoF magnetometer. The onboard firmware runs the necessary sensor fusion algorithms in real time and outputs an accurate estimation of the device's absolute 3D-orientation in an arbitrary fixed reference frame whose z-axis points up. The data is streamed to the host PC with approx. 50 Hz rate via a Bluetooth 4.0 link.

The acquired orientation data is used without additional filtering within the head-finger pointing model (described in Section III) to recover r, which is then intersected with the active workspace surface S to define the pointed-to location.

B. Workspace switching

We define S_{cylinder} as the primary workspace shape (Figure 2a), i.e., the one that is active by default; $S_{\text{h-plane}}$ is considered a secondary shape (Figure 2b), i.e., one that user can switch to upon request (Figure 2c).

The operator switches between the two shapes using a single spring-loaded trigger button on a joystick (Logitech F710); the other joystick buttons and controls are ignored in our experiments. When the trigger is in its default position (not pressed), the primary workspace is used; keeping the trigger pressed uses the secondary workspace. This specific choice is crucial for usability, for two reasons.

First, mapping the explicit state of the button to the chosen workspace shape ensures that the operator is kept aware of the system state—the user must consciously keep the button pressed. On the contrary, toggling the active workspace once the button is released, would make the state implicit (the one the user cannot directly observe).

Second, the trigger functions as a dead man's handle and ensures a fail-safe behaviour by switching the active workspace to the primary (cylinder) configuration when the trigger is released. Since the cylinder workspace is centered

Fig. 3. The hardware used in the experiments: (*left*) Bitcraze Crazyflie 2.0 quadrotor with retro-reflective markers for motion capture system; (*right*) Mbientlab MetaWearR+ IMU bracelet.

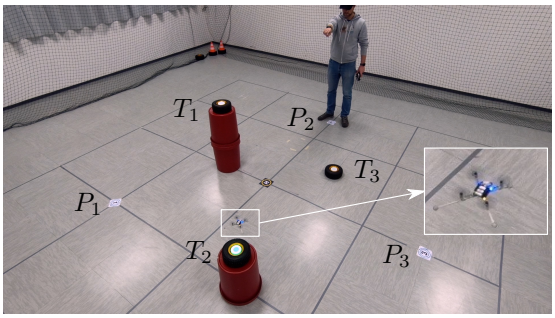

Fig. 4. An overview of the experimental environment during one of the sessions. The LED targets $T_{1,2,3}$ are placed at the wheel hubs at different heights; the user positions are defined at $P_{1,2,3}$.

around the user, it is impossible for the robot to collide with them because of a control mistake. For this reason, the system may also refuse to switch to the secondary workspace if it is considered unsafe: in particular, when the drone is flying close to the height of the user's eyes, the pointing ray is almost parallel to the horizontal plane; in this case, small changes in the arm elevation would result in very large displacements of the pointed location. For this reason, we prevent switching to the secondary workspace if the elevation angle of the drone with respect to the user's head is within $\pm 5^\circ$ from the horizontal direction (see Figure 2d). Whenever a requested switch to the secondary workspace is refused, the joystick vibrates to make sure the operator is notified; a better approach would be to mechanically prevent the trigger button from being pressed when a switch to the secondary workspace would be denied.

C. Flying arena and quadrotor control

Experiments take place in a room with a safety net, outfitted with a commercial optical motion capture system (12 Optitrack PRIME17-W cameras). The Optitrack data is streamed to the Robot Operating System (ROS) with 30 Hz rate. We use a miniature quadrotor Bitcraze Crazyflie 2.0 [34] tracked through a rigid-body marker (Figure 3, left). We also track the location of the user's head through a rigid-body marker attached to a hat. Tracking data is only used for quantitative performance evaluation of experimental runs: the proposed approach is based solely on IMU readings and does not assume that a motion tracking system is available.

D. Human parameters

We configure the kinematic parameters of the human body required by the pointing model before the interaction starts: we set the height of the user's shoulder and head, and the length of the user's arm.

V. Experiments

We conducted an experimental study that evaluates the performance of the proposed 3D-control method against conventional joystick control—the standard interface for manual quadrotor control tasks. Unlike our approach, which implements position control, joysticks operate in the velocity space (one stick controls the vertical velocity and yaw, whereas the other stick controls the velocity along the drone's x and y axes). This is a key difference, which we further discuss in Section VI-A.

A. Setup

We recruited 10 participants (male, mean age 31.1, sd = 5.0) with computer science background, whose goal was to fly the miniature quadrotor over a set of three flat stationary wireless LED beacons (in-house hardware based on Adafruit nRF52 Feather Bluetooth LE board with a ring of 24 RGB NeoPixel LEDs) placed at different predefined heights at known locations (Figure 4).

For each subject we recorded two sessions: one, using the pointing interface (IMU-equipped bracelet on the wrist of their dominant arm); and another, using joystick control (Logitech F710). The order of the sessions was assigned to each subject at random. Each session consisted in three runs. During each run a subject was asked to stand at predefined locations 1–3 and to fly the quadrotor between given targets. The following high-level description of the task was given to participants: "once a target gets illuminated, fly the drone above it".

All subjects reported to be proficient or expert joystick users, and had little or no previous experience with the proposed pointing interface. Short "dry run" sessions were performed for all the users to familiarize them with both interfaces and the task.

The experimental session with the pointing interface proceeds as follows: 1) The operator enters the flying arena and stands at the location 1, the quadrotor lies on the floor at the center of the room; 2) The operator presses once the switch on the bracelet to take off the drone, and second time to initiate the interaction; 3) The operator then points at the drone and holds his arm still for 3 seconds; 4) The bracelet vibrates, the drone makes a small "jump" to show that it is now attached to the operator; the workspace is initialized with the primary surface (cylinder); 5) One of the targets lights up in blue; 6) The operator directs the robot to the target, while when necessary switches the active workspace to secondary surface (horizontal plane); once within the confirmation zone (10 cm from target's center in horizontal plane and 20 cm in along z-axis), the target turns yellow and

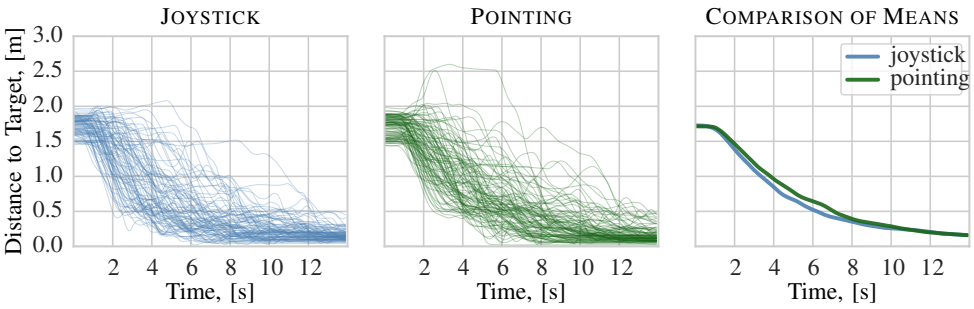

Fig. 5. Analysis of the evolution in time of the distance to the target, for each trajectory flown; each trajectory is represented as a line; $t = 0$ on the plot corresponds to the t_0 time of each trajectory. Left: joystick interface ($N = 90$). Center: pointing interface ($N = 90$). Right: average over all trajectories for each interface.

a timer is triggered; 7) To clear the target the subject must keep the robot within the confirmation zone for 2 seconds. Then, the target shortly turns green and switches off; if the robot leaves the confirmation zone before the two seconds expire, the target turns blue and the timer is reset; 8) Once a target is cleared, the next target lights up in blue: steps 5–7 are repeated; 9) Once all targets are cleared, the operator directs the quadrotor to the landing spot and lands it there by holding their arm still for 3 seconds; 10) The operator moves to the next numbered location and the steps 2–9 are repeated until all three runs are completed.

The procedure for experiments with the joystick is similar, but omits steps 3 and 4 of the above sequence (selection of the quadrotor and switching of the workspace surfaces), which are specific to the pointing interface.

B. Data collection

To assess the performance of the system, we collected the ground truth positions of the participants and the quadrotor, and the times of state transition events of the targets. In our analysis, we ignore parts of the trajectories from the moment the operator takes control till the moment the *first* target is cleared and from the moment the *last* target is cleared till the landing as they do not represent the actual task. We split the resulting trajectories in three segments. Each segment represents a part of the trajectory between two targets. This yields a total of 90 segments for each modality (pointing, joystick), i.e. three segments per run per operator per location. We collect the data with a standard ROS tool `rosbag` and analyze it offline using Python.

To compare the performance of the two interfaces we use the *time-to-target* and the *trajectory length* metrics, which are applied per segment of the flight path.

VI. Results

On Figure 5 we report an evolution of the distance to target in time; the comparison of means on the right plot shows that the performance of both interfaces is very similar. These results also confirm our previous finding for a similar 2D task [2].

To find exact differences in performance of the two interfaces, we further analyze *relative lengths* of trajectories flown with each interface. This measure is calculated as a ratio between the length of a segment flown and the length of an ideal trajectory flown by an optimal controller, i.e. a straight distance between corresponding targets. The closer the relative length to 1.0, the better the performance of a given interface. Figure 7 shows these data on a 2D plot, where median relative lengths for pointing and joystick interfaces define the x and y coordinates of black dots and their mean values define blue crosses. The relative length of trajectories flown with pointing is shorter than the one for joystick interface.

In Table I, we present a detailed comparison of median and mean trajectory lengths (relative to straight line) and median and mean times to target for the two interfaces. For each subject (rows) we report the median and mean performance over all segments, for each of the two control modalities (columns). Since the samples are matched, i.e. we can compare the performance of the same subject on both control modalities, the hypothesis that pointing yields better performance than joystick—lower length or duration—can be assessed for statistical significance using a *paired* difference test. Because the sample size is small and the population cannot be assumed to be normally distributed [35], we use the Wilcoxon signed-rank test [36] as an alternative to the paired Student's t-test. We found that pointing yields shorter trajectories ($p = 0.046$) than joystick; the impact of the control mode on trajectory duration is not statistically significant, even though median duration is slightly better for pointing (14.4 s) than for joystick (15.4 s).

A. Discussion

The shorter lengths obtained with the pointing interface can be explained by smoother trajectories. In fact, visual comparison of trajectory segments performed with the two interfaces (Figure 6) shows that joystick trajectories are comprised of multiple orthogonal pieces in both vertical and horizontal planes, while for pointing that is true only for vertical segments. On the other hand, the time to target metric did not show statistically significant improvement. These results could be due to several reasons.

First, with the pointing interface the operator directly indicates the target position, meaning that low-level trajectory

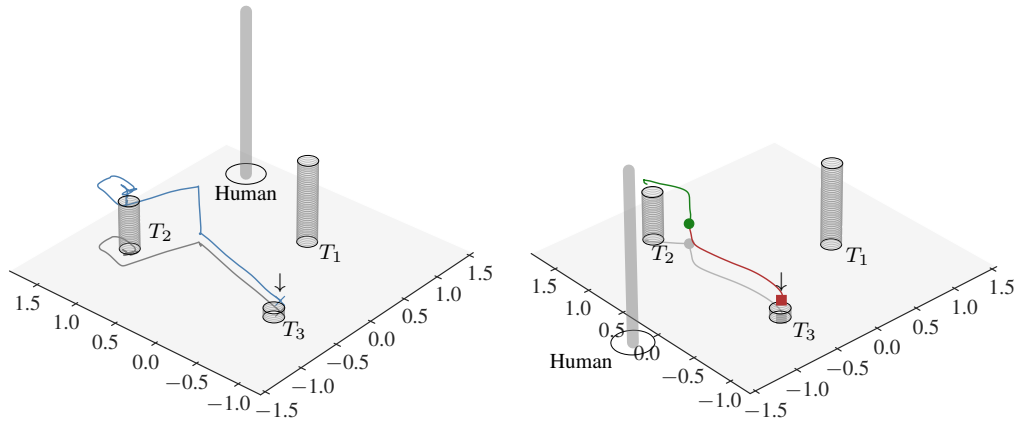

Fig. 6. Visualization of segments of a trajectory performed with joystick (*left*) and pointing (*right*), where the green part of the trajectory is performed using S_{cylinder} and the red one using $S_{\text{h-plane}}$. The short cylinders represent the targets, the tall thin cylinder represents the user, and the arrow is the start of the trajectory.

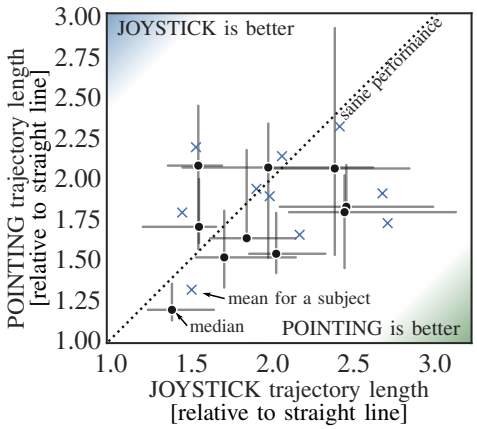

Fig. 7. Relative lengths of the trajectories flown with the pointing and the joystick interface, normalized by straight distances between targets. The black dots and the blue crosses represent median and mean values respectively for each user ($N_u = 10$). The error bars represent 25-th to 75-th percentile.

execution (e.g. deceleration when approaching the target) is performed automatically and more efficiently. On the contrary, with the joystick interface operators give commands in velocity space and have to plan the trajectory themselves. This leads to a sub-optimal control and results in longer trajectories.

Second, quadrotor control with pointing requires more attention from the operator because they cannot "pause" the interaction: once they are attached to the drone they should be careful with the movements of their pointing arm as it immediately translates into the movements of the robot. On the contrary, joystick interface allows to leave the controls at any moment—the drone will just stop and hover. Our intuition is that these differences lead to different strategies adopted by the participants: with pointing they tend to be precise and slow, while with joystick more aggressive, resulting in trial-and-error approach with fast and less precise commands.

TABLE I

PERFORMANCE PER SUBJECT (MEDIAN AND MEAN OVER ALL SEGMENTS)

	Trajectory length, [relative to straight line]				**Duration, [s]**			
Subject	JOYSTICK		POINTING		JOYSTICK		POINTING	
	med	mean	med	mean	med	mean	med	mean
1	1.72	**1.92**	**1.53**	1.95	11.6	**12.7**	**10.4**	14.9
2	**1.57**	**1.46**	1.72	1.81	16.8	18.2	**16.5**	**18.1**
3	**2.00**	**2.08**	2.08	2.15	13.3	13.9	**11.5**	**12.7**
4	2.48	2.70	**1.84**	**1.92**	25.8	26.2	**15.7**	**15.9**
5	2.47	2.73	**1.81**	**1.74**	19.8	18.6	**14.8**	**16.4**
6	1.40	1.52	**1.21**	**1.33**	14.4	14.8	**11.9**	**11.8**
7	2.41	2.44	**2.08**	**2.34**	**15.6**	**16.0**	22.2	21.8
8	**1.56**	**1.55**	2.09	2.21	**10.2**	**10.7**	11.8	14.1
9	1.86	2.00	**1.65**	**1.91**	**11.1**	**12.3**	14.9	15.5
10	2.04	2.19	**1.55**	**1.67**	15.6	15.4	**14.0**	**15.2**

The better result of the two interfaces is shown in bold.

VII. CONCLUSIONS

We presented an intuitive interface for controlling a drone in 3D space using pointing. Although, this control mode has only two degrees of freedom it can be efficiently used when combined with dynamic geometrical constraints imposed by switching workspace surfaces. We have shown that intersecting a pointing ray with such surfaces allows one to efficiently control the robot in the entire 3D space. In particular, a combination of a cylindrical and a planar shape allows operators to precisely control a quadrotor to fly above targets located at different heights.

The current work focused on assessing the operators' performance who used the pointing interface for the first time. We think that, after using the interface for an extended period, operators may perform significantly better using pointing than using a joystick, as already demonstrated by a few more expert users.

ACKNOWLEDGMENTS

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

References

[1] S. Kita, Ed., *Pointing: Where language, culture, and cognition meet*. Manwah, NJ: Lawrence Erlbaum Associates, 2003.

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

[3] W. S. Ng and E. Sharlin, "Collocated interaction with flying robots," *Proceedings - IEEE International Workshop on Robot and Human Interactive Communication*, pp. 143–149, 2011.

[4] K. Pfeil, S. L. Koh, and J. LaViola, "Exploring 3d gesture metaphors for interaction with unmanned aerial vehicles," *Proceedings of the 2013 international conference on Intelligent user interfaces - IUI '13*, p. 257, 2013.

[5] M. Obaid, F. Kistler, G. Kasparaviciute, A. E. Yantac, and M. Fjeld, "How would you gesture navigate a drone?: a user-centered approach to control a drone," in *Proceedings of the 20th International Academic Mindtrek Conference on - AcademicMindtrek '16*, 2016, pp. 113–121.

[6] G. Butterworth, *Pointing: Where language, culture, and cognition meet*. Manwah, NJ: Lawrence Erlbaum Associates, 2003, ch. Pointing is the royal road to language for babies, pp. 9–33.

[7] J. R. Cauchard, J. L. E. Kevin, Y. Zhai, and J. A. Landay, "Drone & Me: An Exploration Into Natural Human-Drone Interaction," *UbiComp '15*, pp. 361–365, 2015.

[8] D. Droeschel, J. Stückler, and S. Behnke, "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.

[9] B. Großmann, M. R. Pedersen, J. Klonovs, D. Herzog, L. Nalpantidis, and V. Krüger, "Communicating Unknown Objects to Robots through Pointing Gestures," in *Advances in Autonomous Robotic Systems 15th Annual Conference, TAROS 2014*. Birmingham: Springer, 2014, pp. 209–220.

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

[11] A. J. B. Trevor, J. G. Rogers, A. Cosgun, and H. I. Christensen, "Interactive object modeling & labeling for service robots," *ACM/IEEE International Conference on Human-Robot Interaction*, p. 421, 2013.

[12] J. Sugiyama and J. Miura, "A wearable visuo-inertial interface for humanoid robot control," in *ACM/IEEE International Conference on Human-Robot Interaction*. IEEE, mar 2013, pp. 235–236.

[13] M. T. Wolf, C. Assad, M. T. Vernacchia, J. Fromm, and H. L. Jethani, "Gesture-based robot control with variable autonomy from the JPL BioSleeve," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 1160–1165, 2013.

[14] A. Jevtić, G. Doisy, Y. Parmet, and Y. Edan, "Comparison of Interaction Modalities for Mobile Indoor Robot Guidance: Direct Physical Interaction, Person Following, and Pointing Control," *IEEE Transactions on Human-Machine Systems*, vol. 45, no. 6, pp. 653–663, 2015.

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

[16] S. Pourmehr, V. Monajjemi, J. Wawerla, R. Vaughan, and G. Mori, "A robust integrated system for selecting and commanding multiple mobile robots," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 2874–2879, 2013.

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

[18] A. Giusti, J. Nagi, L. Gambardella, and G. A. Di Caro, "Cooperative sensing and recognition by a swarm of mobile robots," *IEEE International Conference on Intelligent Robots and Systems*, pp. 551–558, 2012.

[19] Z. Zivkovic, V. Kliger, R. Kleihorst, A. Danilin, B. Schueler, G. Arturi, C.-C. Chang, and H. Aghajan, "Toward low latency gesture control using smart camera network," in *Computer Vision and Pattern Recognition Workshops, 2008. CVPRW'08. IEEE Computer Society Conference on*. IEEE, 2008, pp. 1–8.

[20] K. Plaumann, M. Weing, C. Winkler, M. Müller, and E. Rukzio, "Towards accurate cursorless pointing: the effects of ocular dominance and handedness," *Personal and Ubiquitous Computing*, pp. 1–14, 2017.

[21] S. Mayer, V. Schwind, R. Schweigert, and N. Henze, "The Effect of Offset Correction and Cursor on Mid-Air Pointing in Real and Virtual Environments," *Proc. of the 2018 CHI*, 2018.

[22] K. Nickel and R. Stiefelhagen, "Pointing Gesture Recognition based on 3D-Tracking of Face , Hands and Head Orientation Categories and Subject Descriptors," *Proceedings of the 5th international conference on Multimodal interfaces*, pp. 140–146, 2003.

[23] S. Ueno, S. Naito, and T. Chen, "An efficient method for human pointing estimation for robot interaction," in *2014 IEEE International Conference on Image Processing (ICIP)*. IEEE, oct 2014, pp. 1545–1549.

[24] S. Mayer, K. Wolf, S. Schneegass, and N. Henze, "Modeling Distant Pointing for Compensating Systematic Displacements," in *Proc. of the ACM CHI'15*, vol. 1, 2015, pp. 4165–4168.

[25] K. Kondo, G. Mizuno, and Y. Nakamura, "Analysis of Human Pointing Behavior in Vision-based Pointing

Interface System - difference of two typical pointing styles -," *IFAC-PapersOnLine*, vol. 49, no. 19, pp. 367–372, 2016.

[26] R. A. Bolt, ""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.

[27] R. Jota, M. a. Nacenta, J. a. Jorge, S. Carpendale, and S. Greenberg, "A Comparison of Ray Pointing Techniques for Very Large Displays," *GI '10 Proceedings of Graphics Interface 2010*, pp. 269–276, 2010.

[28] A. Cockburn, P. Quinn, C. Gutwin, G. Ramos, and J. Looser, "Air pointing: Design and evaluation of spatial target acquisition with and without visual feedback," *International Journal of Human Computer Studies*, vol. 69, no. 6, pp. 401–414, 2011.

[29] O. Erat, W. A. Isop, D. Kalkofen, and D. Schmalstieg, "Drone-Augmented human vision: Exocentric control for drones exploring hidden areas," *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 4, pp. 1437–1446, 2018.

[30] J. S. Pierce, A. S. Forsberg, M. J. Conway, S. Hong, R. C. Zeleznik, and M. R. Mine, "Image plane interaction techniques in 3D immersive environments," *Proceedings of the 1997 symposium on Interactive 3D graphics - SI3D '97*, pp. 39–ff., 1997.

[31] M. Karrer, P. Schmuck, and M. Chli, "Cvi-slam—collaborative visual-inertial slam," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 2762–2769, 2018.

[32] B. Gromov, L. Gambardella, and A. Giusti, "Robot identification and localization with pointing gestures," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2018, to appear.

[33] Mbientlab official web-page. https://mbientlab.com/.

[34] Bitcraze, "The crazyflie nano quadcopter," https://bitcraze.io.

[35] R. Lowry. Concepts and applications of inferential statistics. http://vassarstats.net/textbook/. [Online; accessed: 2019-08-26].

[36] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bulletin*, vol. 1, no. 6, pp. 80–83, 1945.