Fine grained pointing recognition for natural drone guidance

O. L. Barbed P. Azagra
DIIS-i3A
University of Zaragoza, Spain

L. Teixeira M. Chli
Vision for Robotics Lab
ETH Zurich, Switzerland

J. Civera A. C. Murillo
DIIS-i3A
University of Zaragoza, Spain

Abstract

Human action recognition systems are typically focused on identifying different actions, rather than fine grained variations of the same action. This work explores strategies to identify different pointing directions in order to build a natural interaction system to guide autonomous systems such as drones. Commanding a drone with hand-held panels or tablets is common practice but intuitive user-drone interfaces might have significant benefits. The system proposed in this work just requires the user to provide occasional high-level navigation commands by pointing the drone towards the desired motion direction. Due to the lack of data on these settings, we present a new benchmarking video dataset to validate our framework and facilitate future research on the area. Our results show good accuracy for pointing direction recognition, while running at interactive rates and exhibiting robustness to variability in user appearance, viewpoint, camera distance and scenery.

1. Introduction

Human-machine interaction, in particular intuitive user-interface, is a critical aspect to be addressed before emergent technologies can be massively adopted by the society. Human action recognition systems typically classify user gestures into different types of actions, e.g., pointing vs. waving. Differently, our work focuses on fine grained analysis of pointing gestures, to enable more natural interactions to guide robots' motion. Existing interfaces to control robotic platforms such as Unmanned Aerial Vehicles (UAVs) remain largely unintuitive to date, often requiring extensive training of expert pilots. Hand-held radio transmitter panels or mobile phones are most commonly used to pilot UAVs today, and both options occupy both hands of the pilot constantly. Realizing this limitation, the most prominent gesture-based command implemented in commercial drones currently is one to capture selfies, available for instance on some DJI drones. More natural ways of human-UAV interaction are needed for recent advancements to be leveraged in reality and adopted by users.

Figure 1. Interaction by natural pointing. This work explores how to command a drone with natural human gestures (by pointing to the direction of the desired motion).

The goal of the presented human-UAV interaction system is to command the UAV motion by processing a video of the user pointing towards the desired motion direction, as represented in Fig. 1. Our approach does not only provide an intuitive way of piloting an intelligent UAV, but also promises to release both user hands, enabling the user to continue with other tasks (e.g. search-and-rescue) instead of constantly monitoring the drone's trajectory. We build three pointing direction recognition strategies based on the state-of-the-art related to the three most common approaches for people analysis: person segmentation, skeleton keypoints detection, and specific body part detection, namely face and hands. This work evaluates a broad set of variations, from deep learning models to simpler heuristics and classifiers, to decide the most suitable strategy for the presented goals.

Besides, to the best of our knowledge there is no existing benchmark for the nature of interactions targeted in this work, so we have built and released a new dataset [1]. This dataset consists of videos of users facing the camera giving motion directions by pointing, incorporating challenges such as increasing camera-user distance and large user and scenario variability. The evaluation of the proposed approach on this benchmarking dataset demonstrates the promising performance of our proposal, exhibiting robustness to the user, scenario and camera variability, while maintaining the capability to run at interactive rates. The processing can be done on a laptop (Nvidia GTX 1070) as base station at 4.5 fps or onboard the robot (Nvidia Jetson AGX Xavier) at 0.61 fps.

[1] https://sites.google.com/a/unizar.es/hri-drones/

2. Related Work

The most relevant related topics for our work are human detection and pose analysis and Human-Robot Interaction, in particular interaction with drones.

2.1. Human detection and Pose/Gesture recognition

Visual human detection and pose and gesture classification are all extensively explored problems but still with significant challenges ahead. As in many other computer vision tasks, deep learning-based solutions have boosted the performance of person detection algorithms. We find two extensive groups of approaches. The first group consists of target object detection (including people) approaches, that provide a bounding box surrounding the objects, such as the very efficient YOLO by Redmon et al. [18]. This is the base of one of our strategies, which builds on hands and face detection on video frames. The second group of approaches is related to recent advances on semantic segmentation, and assigns a class label to every pixel in the image (including people as one of the most common classes), e.g., the well-known He et al.'s Mask-RCNN [7]. This is the base for two of our strategies as, in addition to its excellent performance, it has a versatile architecture that can be used to segment people but also to estimate human pose and skeleton detection. We will use both outputs in our approach.

There is extensive work in the literature about human posture analysis. Several approaches use 2D estimations, such as Insafutdinov et al. [8] or Cao et al. [2], and are based on deep learning and the use of confidence for each of the joints. Other works like Mehta et al. [11] create a 3D pose estimation representation, extending the use of confidence heat maps to 3D matrices. For the 3D case, we also find prior work that directly includes depth information for the posture or gesture analysis, like the work done by Molchanov et al. [13]. Rather than focusing on the detailed human pose, other works focus on recognizing specific human gestures, as a goal in and of itself, e.g. in deaf gesture-recognition by Cui et al. [3], or as part of human-robot interaction applications, e.g. in Azagra et al. [1]. Differently from these works, ours does not target recognizing different gestures nor finding the object a user finger is pointing, but using the common gesture of *pointing* to give navigation commands.

2.2. Human-Robot interaction

There are plenty of strategies for human-robot interaction, depending among others on the sensor modalities or the specific interaction types. Strategies based on visual gesture recognition and hands-free machine-human interaction have been widely explored since the emergence of RGB-D cameras and its popularization, typically focusing on ground robots. For example [14] is an early work on pointing direction estimation for commanding ground robots.

In the recent years and closer to our goals, since recently smaller and cheaper drones are available to general public, human-drone interaction has received more attention. We can refer to works like Hansen et al. [6], where they use the user's gaze, recorded at a base station, to guide the drone, or multimodal works like Fernandez et al. [4], that uses several modalities (voice, gesture, QR codes or GUI) to guide the drone. Other works like Knierim et al. [9] use a small drone in a cage with which the user is able to interact tangibly. [17] uses a classification scheme based on skeleton detection to command the robot up and down. However, it uses a RGB-D camera, which limits the drone location. The experimental evaluation is also limited to a proof of concept. [10] proposes the use of future regression networks for early classification of distinctive human gestures into a set of different actions (take a selfie, move or stop). The work done by MohaimenianPour et al. [12] is closer to our goals. Authors apply visual gesture recognition to human-drone interaction. Its approach is based on a set of pre-defined configurations of the relative location of the hands and face recognized in the image, and each configuration corresponds to a pre-defined action. However, the study done by Obaid et al. [15] shows that users are more prone to do *deictic* interactions with the drone, the type that we consider in our approach, which are more natural and intuitive to the user than pre-defined ad-hoc gestures.

The work most similar to ours [21] also presents an algorithm to command a robot by pointing. We built a similar strategy, based on the detection of hands and faces, but using a more recent detector, in addition to our two other novel approaches based on skeleton detection and semantic segmentation.

3. Drone Guidance by Natural Pointing

3.1. Overview

As summarized in Fig. 2, our pipeline for high-level drone guidance through natural pointing interactions has the following components:

1) Input. We consider two input setups, the camera is placed on board the robot (to enable configurations where the robot operates near the human user) or at a base station (if the robot operates at scenarios not accessible for the human).

2) Pointing direction recognition. This is the core component of our pipeline and consists of three stages.

Person detection and representation (per frame). We detect the persons in every frame, exploring three alternatives detailed in section 3.2.

Direction Classification (per frame). To estimate the direction where the person detected is pointing, we discretize

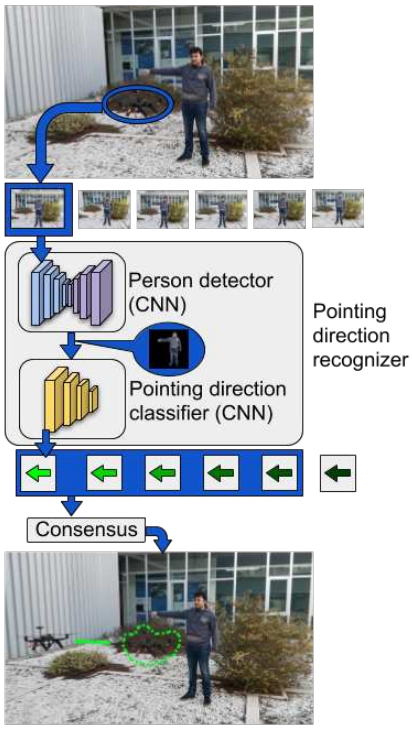


Figure 2. Drone guidance by natural pointing. Our system receives the captured video as input, estimates the user pointing direction seeking consensus across a window of frames and sends the command to the drone using ROS.

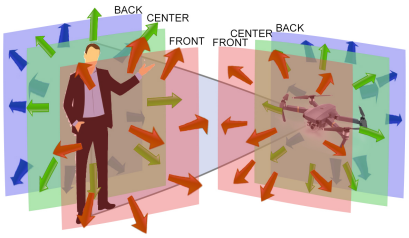
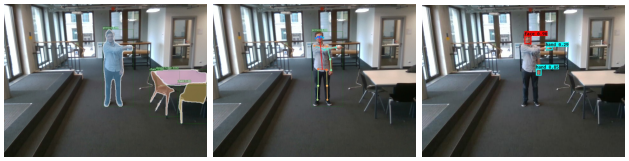


Figure 3. Illustration of the 26 navigation directions recognized by our system. In the experiments where only 8 directions are named, we use the CENTER plane directions.

the space of possible navigation directions (see Fig. 3) and formulate this step as a classification problem. The alternatives explored are also detailed in section 3.2.

Consensus (over temporal window). The classification of natural gestures addressed in this work is challenging due to the high variability and limited training data. We apply a voting scheme over several frames within a small sliding window (5 frames in our experiments) as an effective way to add robustness to our classification. This consensus block receives, from the previous block, the estimated pointing direction and a confidence value for each frame within the window. It only accepts a command if 4 of the



(a) (b) (c)

Figure 4. Example of the three alternatives for person detection and representation. (a) Segmentation. (b) Skeleton. (c) Hands&Faces

5 last images agree on the command. If the confidence on the classification of a frame is below certain threshold (0.5 in our experiments), the vote of that frame is ignored.

3) Output (Drone command). Once there is consensus on a certain direction, the corresponding command is sent to the real or simulated UAV via network message. The message contains the x, y and z coordinates of the relative position where the drone has to move to.

3.2. Pointing direction recognition in RGB

This subsection details the recognition of the user pointing direction, the core component of our pipeline.

The first step is to detect the persons in the scene, to select the region of interest (ROI) and represent the data adequately for the direction classification stage. Regarding this final direction classification step, there are two main constraints to consider. First, due to the interactive application targeted, the system needs to react to the user actions in acceptable rates for an interactive application (*i.e.*, a few milliseconds). Second, since the amount and heterogeneity of the labelled data available is fairly small (see section 5 for a detailed description of the data used), we need to consider simple models or transfer learning techniques, rather than training from scratch large models.

Following the three most common alternatives in the literature for people detection or segmentation in images, we have built the three strategies detailed next (illustrated in Fig. 4): *Segmentation*, *Skeleton* and *Hands&Faces*.

Segmentation. This strategy is based on a well known semantic segmentation CNN, Mask R-CNN [7], to detect all the people in the scene (we use the official implementation, Detectron [5] pre-trained on the COCO dataset).In particular we use a publicly available model for scene segmentation[2]. This model labels every pixel in the image with one of the target labels (*i.e.*, semantic/instance segmentation), one of them being *person*. From this output, we keep the image segments with the *person* label.

The person segment with the largest area is designated as the pilot. We select the minimum-size squared patch that contains the pilot segment. We also mask out the pixels

[2]Set 12_2017_baselines, model e2e_mask_rcnn_R-101-FPN_2x [5]

that do not belong to the person. With this we remove irrelevant background information that could have a negative influence in next stages.

For the direction classification part, we have explored several CNN architectures for image classification, offering MobileNetV2 [20] the best compromise between performance and delay. MobileNet is a well known efficient architecture very well suited for applications with execution time restrictions.

Skeleton. This strategy is also based on Mask R-CNN [7], but in this case we use a model pre-trained to estimate person skeleton keypoints[3]. This model estimates the postural information from all the people in the image. In particular it provides a list of the coordinates of the following keypoints for each person found: {*nose, left eye, right eye, left ear, right ear, left shoulder, right shoulder, left elbow, right elbow, left wrist, right wrist, left hip, right hip, left knee, right knee, left ankle, right ankle*}. These 17 keypoint coordinates are used to represent a person. As in the previous strategy, we designate the person of largest area as the pilot.

To recognize the pointing direction, we first calculate the angle (θ) of the arm link between the elbow and the wrist, using the corresponding skeleton keypoints p_{elbow} and p_{wrist} coordinates:

$$\theta = \text{atan2}\left(\frac{p^y_{elbow} - p^y_{wrist}}{p^x_{elbow} - p^x_{wrist}}\right). \quad (1)$$

θ is computed for both arms and the system chooses the arm that is farther from the "resting" position as the "pointing" arm. We have explored numerous options to classify θ into one of the 8 possible pointing directions. The most relevant are the following:

- **Nearest Neighbour (NN).** The median of the orientation of the arm link for each class in the training examples is computed. Given a new θ, it is assigned to the class of the closest median. This is a fairly simple process, which does not contemplate the possibility of having an *unknown* class.

- **SVM.** Standard RBF-kernel SVM classifier [16]. Different kernel functions and configurations have been evaluated and the RBF kernel obtained the best performance.

- **Decision-Tree.** Standard Decision Tree classifier [16]. We also considered compound classifiers like Random Forests, but they converged to a single tree because of the simplicity of the input (a single number/angle).

[3]Set 12_2017_baselines, model e2e_keypoint_rcnn_R-101-FPN_1x [5]

Table 1. Detection results of the *Hands&Faces* YOLOv3 model on our dataset.

Result found	DDIR-1	DDIR-2
No face	133 (7.95%)	720 (22.41%)
Only face	97 (5.80%)	75 (2.33%)
Face and one hand	698 (**41.75%**)	668 (**20.8%**)
Face and two hands	744 (**44.50%**)	1749 (**54.45%**)
Total	1672	3121

Hands&Faces. This third strategy is based on the detection of the person's hands and face, rather than segmenting the whole body. It is inspired by [12], where they fine-tuned a YOLOv2 [18] model, with a new dataset they released, to detect hands and faces. Using their released dataset, we have fine-tuned a COCO-pretrained YOLOv3 [19] model. This model outputs the position and size (*i.e.*, bounding box) of the hands and faces that appear in the image.

We evaluate the detector obtained on both challenges presented by the hands and faces data authors [12], for hands[4] and faces[5] detection respectively. We obtain an AP of 87.7 and AR of 74.5 for the VIVA hand detection challenge and AP average of 0.36 in the WIDER face challenge. These results show our model does not reach the top performance in the face detection, however we should note that the challenge poses very general and heterogeneous face detection tasks, which are often far from the type of images we expect in our system. Analyzing the detection results of the obtained model in our data, shown in Table 1, we observe that it detects at least a hand and a face (enough for our approach to work) in most of the images (75-80%).

Inspired by [12], the relative position of the hands and face is computed in our algorithm to identify the pointing direction. We approximate the chest position based on the detected face and trace a ray to the center of the hand, as an approximation to the arm pointing direction. This way the pointing angle is computed similarly to previous approach:

$$\theta = \text{atan2}\left(\frac{p^y_{chest} - p^y_{hand}}{p^x_{chest} - p^x_{hand}}\right). \quad (2)$$

θ is again computed for both hands and the one farthest from the "resting" position is kept. Once the angle of the link between a hand and the chest is computed, the classification of the pointing direction is computed in an identical manner to the previous approach.

There are significant differences between the three strategies we have built. The biggest advantage in the *segmentation* is that it works with the input image directly, which means that our objective of making this a natural pointing recognition system is easier since we do not have to define the gestures with posture geometry. However, since

[4]http://cvrr.ucsd.edu/vivachallenge/index.php/hands/hand-detection/

[5]http://mmlab.ie.cuhk.edu.hk/projects/WIDERFace/WiderFace_Results.html

it is a more variant representation, the success of this strategy depends greatly on the heterogeneity of the training dataset to make sure it generalizes correctly. The *skeleton* and *Hands&Faces* representations are far more abstract and invariant to the person and their surroundings, and consist of much smaller descriptors, which facilitate an efficient classification. As we mentioned, in these approaches the gestures need to be defined with posture geometry and they discard the visual information from the image, so any error in the skeleton information has much more effect on the results.

3.3. Pointing direction recognition in RGB-D

As a more general extension to the 2D pointing direction recognition task, we also explored how to make the system recognize 3D pointing directions. The added value of this extension is evident due to the increase in maneuverability.

We recorded and labeled the subset DDIR-5 which contains RGB-D images of users performing 26 different 3D pointing gestures (the 3D directions considered are represented in Fig. 3). We consider the same 8 pointing directions than the 2D case, but in three different depth planes: center (aligned with the person), front (closest to the camera) and back (furthest from the camera). We attempted to classify all the 3D directions on an end-to-end similar to the 2D case, but as we detail later in the experiments, the best option is to separate the 2D direction and the depth classification problems. We solve this with an additional classifier to identify the depth. We discretized the depth values into three possible classes: *back*, *center* and *front*, corresponding to the space in front of the user (*front*), at the same depth that the user (*center*) or the plane behind the user (*back*).

The approach used for this additional module is based on the *skeleton* approach. We detect the skeleton keypoints using the *skeleton detector* and use the x, y and depth from those points to calculate the x-, y- and z-angle of the vector that goes from the centroid of the skeleton to each keypoint.

4. DDIR Dataset

As there is no prior published data on the problem we address, we have built and release a new dataset[6], **Direction Dataset for Interaction with Robots (DDIR)**, that we are releasing to the community. The data is organized in five subsets (DDIR 1 to 5) depending on different characteristics. Fig. 6 shows sample frames of each of these five sets, which are detailed next. The data has been labeled with the corresponding 2D pointing direction out of a 8-bin representation (except DDIR-5 which has 26 possible directions in 3D) and an *unknown* class label. Fig. 5 shows representative examples of the dataset classes and Table 2 summarizes the data technical specifications.

[6] https://sites.google.com/a/unizar.es/hri-drones/

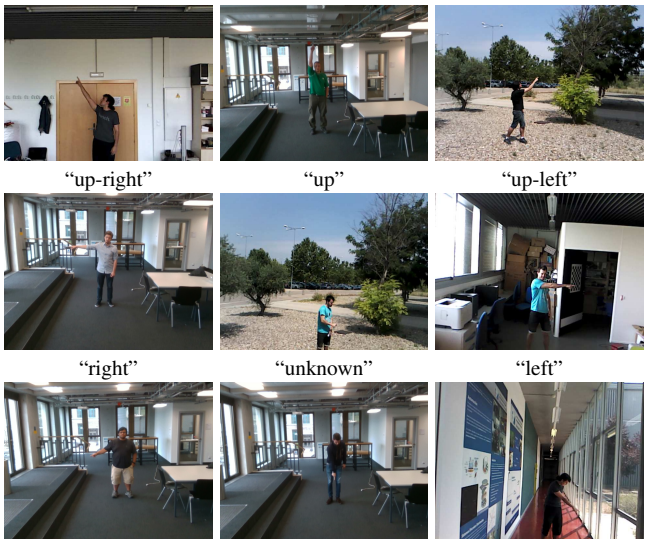


Figure 5. Example of each 2D direction class considered in our data. The label of each class (right under each image) is the direction in which the person is pointing. The "unknown" class is the most heterogeneous because it covers every image in which the pointing direction is not clear (or the user is not pointing).

Table 2. Summary of the presented dataset.

	DDIR-1	**DDIR-2**	**DDIR-3**	**DDIR-4**	**DDIR-5**
Image resolution	VGA	VGA	VGA	FWVGA	FWVGA$^+$
# users	5	3	6	5	7
# actions per user	8*	8*	48*	64*	52*
User distance (m)	5	2.5-5	2.5-10	5-10	2.5-5
# Indoor/Outdoor scenarios	1/0	3/1	2/1	0/2	4/3
# direction classes	8	8	8	8	26
# frames, direction classes	1393	3212	16430	11711	22628
# frames, *unknown*-class	2035	2093	6356	16520	4726

* Evenly distributed for each class. $^+$ RGB-d recording.

DDIR-1. This set was recorded at an indoor scenario (location 1) with the camera plugged into the base station. It is used for training and testing, splitting the data following a cross-validation strategy. For each fold we always leave one user data out of the training set (*DDIR-1, train-fold:* images from 4 of the 5 users. *DDIR-1val, validation-fold:* images of the remaining user used for the validation).

DDIR-2. This set was recorded at three indoor and one outdoor scenarios (at location 2, a different location that DDIR-1), also with the camera plugged into the base station. This data is used to evaluate robustness to scene and user changes. The domain change is challenging but allows us to demonstrate how well the algorithms generalize.

DDIR-3. This set was recorded at three different scenarios: two indoors and one outdoors (at location 2). The different users perform the pointing gestures 2.5, 5 or 10 metres away of the camera. This dataset is used together with the next one for testing the complete system.

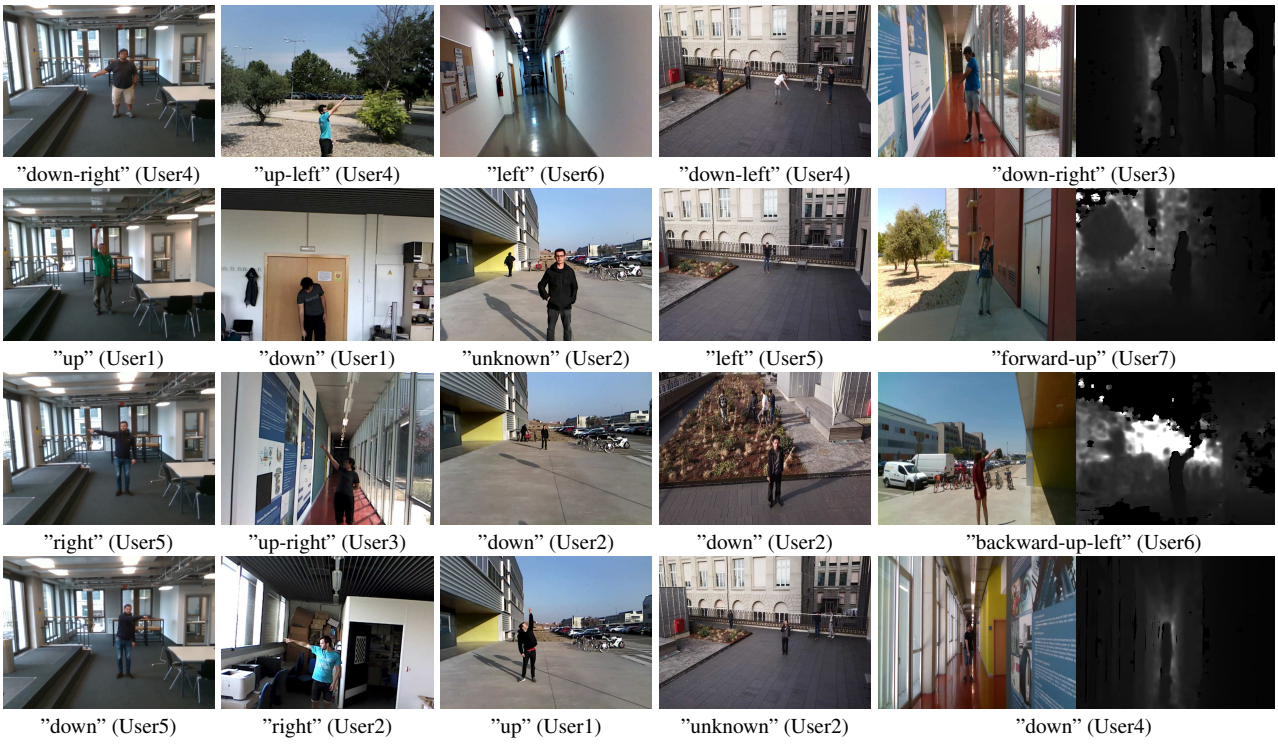


Figure 6. Examples of each of the five sets of the presented *Directions Dataset for Interaction with Robots* (**DDIR**). Each example is described with the direction and the user performing the action

DDIR-4. This set was recorded in two different outdoor scenarios (at location 1). In this case the camera is aboard the drone, and the drone is hovering at 3 metres above the ground. The different users perform the pointing gestures 5 and 10 metres away from the drone. In half of the footage there are other people in the background while the user is pointing. This data is essential to demonstrate robustness to different image viewpoints due to the fact of having the camera on the base station or attached to the drone.

DDIR-5. This set was recorded in seven different scenarios: four indoors and three outdoors (at location 2). The users perform gestures at approximately 5 metres from the camera. The camera used in this set has infrared sensors that let us also record depth information of the footage. This set is used for expanding the system to 3D movement, so the pointing directions cover all the 26 shown in Fig. 3. For this set we split the data similarly to set DDIR-1: DDIR-5 is the *train-fold* and DDIR-5val is *validation-fold*.

5. Experimental Results

We analyze the different alternatives of our approach and then analyze the performance of our best system configuration. All the experiments were run with an Intel® Core™i7-6700 CPU@3.40GHz×8, 32GB RAM and GPU Geforce GTX 1070 8GB DDR5.

Table 3. Different representations trained on DDIR-1 train set and evaluated on different scenarios.

	Test on: **DDIR-1val**		Test on: **DDIR-2**	
	8 classes	8 classes+ "unknown"	8 classes	8 classes+ "unknown"
(a) Segmentation representation				
MobileNetV2	90.1 (8.2)	68.3 (22.3)	86.5 (12.0)	71.9 (22.9)
MobileNetV2-D	**93.0** (4.9)	**76.2** (18.6)	**90.6** (9.2)	**78.8** (19.2)
(b) Skeleton representation				
NN	93.0 (10.8)	N/A	**91.9** (6.5)	N/A
SVM	95.3 (6.6)	**76.5** (30.6)	90.4 (8.3)	**76.2** (28.1)
Decision-Tree	**95.5** (3.2)	57.8 (18.9)	89.6 (10.0)	57.2 (24.7)
(c) Hands&Faces representation				
NN	54.4 (26.3)	N/A	47.0 (24.5)	N/A
SVM	60.3 (27.0)	**46.6** (33.9)	50.9 (25.7)	**42.9** (28.0)
Decision-Tree	**61.5** (27.7)	40.9 (23.7)	**51.6** (25.5)	32.4 (22.9)
Using only images with a detected face and at least one hand				
NN	71.4 (17.6)	N/A	71.2 (25.1)	N/A
SVM	78.8 (15.4)	**58.3** (33.6)	76.4 (24.1)	**63.5** (31.4)
Decision-Tree	**80.4** (16.0)	51.5 (21.4)	**77.6** (23.6)	46.1 (22.4)

5.1. Analysis of design choices and alternatives

Per frame classification with different representations. For all these experiments, the models were trained using cross validation on the ***train-fold*** from **DDIR-1** set, vali-

dated on the ***validation-fold*** from the same set, and tested on the **DDIR-2** set for additional verification. In order to select the most promising configurations to continue with the complete system evaluation, we computed the recall for each variation, to understand the amount of frames that each method was able to identify.

Table 3(a) corresponds to the **segmentation representation** results. Models were trained, as previously explained, fine-tuning a MobileNetV2 model pretrained on ImageNet. Fine-tuning was run during 100 epochs, with parameter $\alpha = 1.0$ and *learning rate* set to 10^{-4}. A second version (MobileNetV2-D) has been trained using additional data augmentation to account for larger scale varieties, consisting of random image re-sizes from 1:2 to 1:0.5, which achieves better results. Table 3(b) shows the results obtained with variations of the **skeleton representation**. All the alternatives for this representation achieve comparable results, slightly better for the NN, also the simplest to implement.

Table 3(c) shows results for the pointing direction task with our *Hands&Faces* representation strategy. The core component of the **Hands&Faces representation** is the hands and faces detector detailed in section 3.2. Under the same conditions as the other approaches, the NN and the Decision Tree results are very similar, and in both cases significantly lower than the other strategies. As expected, part of this is due to errors in the hands and face detection. If test images where at least a hand and a face are detected are the only ones considered, the results are significantly better but still lower than the other approaches as shown in the same table.

Discussion. We noticed that including an *unknown* class, corresponding to images with non-pointing gesture, drops the performance of our system significantly (around 10% difference between the columns "8 classes" and "8 classes + *unknown*" in all configurations). Therefore, we opted for training only for the 8 direction classes and a different strategy to account for robustness to ambiguous actions (*i.e.*, non pointing). We included the described small temporal consensus stage that filters the classification results in section 3.1.

The results from the Hands&Faces strategy are far from the results from the other two strategies, regarding accuracy and robustness, so we discarded this option for further analysis in the following experiments. Note that the skeleton representation is very compact, which is convenient for efficiency but it may lose useful information such as the appearance. Both the *Skeleton* and *Segmentation* results (in Table 3(a) and (b) val/test columns) show that when the test data is from a domain further to the training one, the average results remain almost intact demonstrating good generalization of our models.

Figure 7. Example of a limitation of our system. For distances larger than 6 meters, our pipeline fails to identify the direction due to the low resolution of the user.

Table 4. Models trained on DDIR-3&4 and tested on DDIR-3&4val which contain data acquired at different distances.

Skeleton-NN	Test at 5m	Test at 10m
Trained at 5m	83.3 (7.6)	70.8 (12.7)
Trained at 5m&10m	83.2 (7.8)	70.7 (12.8)
Segmentation-MobileNetV2-D		
Trained at 5m	89.0 (4.7)	70.9 (6.1)
Trained at 5m&10m	88.4 (6.5)	70.6 (7.6)

Robustness to various camera distances. The two best configurations (Skeleton-NN and Segmentation-MobileNetV2-D) were further evaluated for robustness, on a similar experiment that the one shown in Table 3 but this time training on the DDIR-3&4 training sets and evaluated on DDIR-3&4val, as shown in Table 4. This experiment shows that our system performs best when the user is between 2 and 6 metres from the camera. At larger distances the person is imaged at an extremely low resolution in the cameras we used to record the datasets (see Fig. 7 for two examples). The data augmentation done on the training sets is enough for a model to reach the same accuracy at long distances (more than 6 metres) than models trained directly with data recorded at those distances, as shown in Table 4. These results point that the segmentation approach is more robust to scale changes due to different distances to the camera, therefore it is the most suitable strategy for our system.

5.2. Video classification system

The following experiments evaluate in more detail the best configuration of our system. The per frame classification is combined with a more robust consensus strategy for the final video classification and different aspects of interest are discussed to demonstrate the applicability of this approach.

Consensus strategy benefits. As expected, our whole approach including the consensus stage ("Consensus improvement") obtains better results than "Per frame" classifications. Table 5 shows the results of a more detailed evaluation in a more challenging setup than the preliminary evaluations from previous subsection. Models were trained on DDIR-1&2 data, recorded from a base-station camera,

Table 5. Segmentation strategy trained on DDIR-1&2 and evaluated on DDIR-3 and DDIR-4. Precision-Recall running independent *Per frame* classification (PF) vs applying Consensus (C).

	Per Frame (+ Consensus improvement)			
	DDIR-3		DDIR-4	
Class	Precision	Recall	Precision	Recall
up	63.2(+13.1)	56.1(+10.9)	89.1(+6.4)	34.6(+10.2)
up-right	82.3(+8.3)	65.2(+7.3)	85.7(+6.0)	83.4(+8.7)
right	85.8(+5.8)	63.5(+9.5)	93.0(+2.8)	69.4(+11.4)
down-right	81.7(+10.4)	70.9(+13.6)	69.4(+13.1)	88.6(+4.9)
down	45.3(+14.8)	87.6(+6.0)	68.7(+10.3)	77.8(+11.7)
down-left	83.4(+8.8)	78.6(+9.8)	76.4(+12.4)	87.4(+8.6)
left	79.9(+10.1)	72.7(+11.8)	66.7(+14.3)	76.5(+10.7)
up-left	67.5(+13.8)	70.9(+10.9)	66.6(+20.2)	76.2(+12.8)
Avg PF	73.6	70.7	77.0	74.2
Avg C	**85.3**	**83.2**	**87.7**	**84.1**

and evaluated on DDIR-3 and DDIR-4, where DDIR-4 was recorded from an on-board drone camera. First note the consensus improves around 12% for DDIR-3 and around 10% for the most challenging test of DDIR-4. This experiment results also show that changes in perspective or camera type do not affect the good performance of the system, showing good generalization.

Robustness to user, scenario and camera variations. Besides the robustness to changes in camera type and perspective, the system presents good invariance to all the relevant changes considered with the presented dataset. Robustness to user variations can be analyzed in all experiments since different users appear in all datasets. Even multiple users appearing on the background of several scenes is not an issue for the system, as long as the user providing the command is the closest to the camera (as assumed by the system). It is also relevant to note that training in one environment (DDIR-1&2) and evaluating in a completely different one (DDIR-3 or DDIR-4) provides very good results, see table 5. This demonstrates the good generalization of the model learned to different scenarios.

System Performance. Our final implementation runs the Mask R-CNN for *Segmentation* on the GPU, while simultaneously the *Pointing direction classifier* is run, using MobileNetV2-D, on the CPU.

The processing time of one image, running our whole system, is an average of 225ms (the detector takes more than 90% of the time). This means that the proposed final system can run at 4.5 fps. The best compromise between usability and accuracy was found using a 5-frame window for the consensus. Longer windows can improve the performance but require the gesture to be performed for longer and it becomes less natural for the user.

In hopes of exploring the use of this system in a drone without a base station, we installed and measured the full system working on a Jetson AGX Xavier. The system takes 1638ms on average to process one image, which means it could run in it independently at 0.61 fps.

The most significant limitations of the current system are the following. As the whole pipeline runs at 4.5 fps and the consensus system requires 5 images to decide, the user has to keep pointing for at least 1 second. While we find it a reasonable time, it also means that quick gestures are not recognized by our pipeline. Besides, the performance decreases with distances larger than 10 meters of the user to the camera. This means that the set up has two possibilities: either the camera is on board the robot with the robot not further than 10 meters from the pilot, or the camera should be placed on a base station near the pilot.

6. Conclusions

Driven by the need for natural and intuitive human-robot interaction, this work proposes a novel framework for providing navigation directions to a drone by a user pointing towards the desired direction. Note that the system could be easily adapted for interaction with any other type of robotic platform. Our system is shown to work both with images from a camera on board the drone or images from a central station camera if drone would be in non-accessible location for a human operator. The proposed approach exhibits both accuracy and robustness to variations in the users, the scenarios and the viewpoints.

To the best of our knowledge, the gesture-based interaction we propose focuses on the under-explored problem of fine-grained analysis of user gestures (pointing in our case), which can significantly help in natural human-robot interaction systems. As part of this work, we release a new dataset for fine-grained pointing recognition, consisting on five different sets imaging different users, scenes and viewpoints. The last one of these sets also includes depth information. There is no similar dataset in the literature, and hence we believe this is a valuable contribution for the community.

For future work, we would like to increase the commanding capabilities by including additional natural gestures in our system. Besides, with our promising experimentation using embedded systems capable of running deep-learning models, we will further investigate transferring all computation on board the drone for increased practicality.

Acknowledgements

The authors would like to thank NVIDIA Corporation for the donation of a Titan Xp GPU used in this work. This research has been partially funded by FEDER/Ministerio de Ciencia, Innovación y Universidades - Agencia Estatal de Investigación under grants PGC2018-096367-B-I00, PGC2018-098817-A-I00 and RTC-2017-6421-7 and Aragón regional government (DGA T45 17R/FSE).

References

[1] Pablo Azagra, Florian Golemo, Yoan Mollard, Manuel Lopes, Javier Civera, and Ana C Murillo. A multimodal dataset for object model learning from natural human-robot interaction. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 6134–6141. IEEE, 2017. 2

[2] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2D pose estimation using part affinity fields. In *CVPR*, 2017. 2

[3] Runpeng Cui, Hu Liu, and Changshui Zhang. A deep neural framework for continuous sign language recognition by iterative training. *IEEE Transactions on Multimedia*, 2019. 2

[4] R. A. S. Fernandez, J. L. Sanchez-Lopez, C. Sampedro, H. Bavle, M. Molina, and P. Campoy. Natural user interfaces for human-drone multi-modal interaction. In *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 1013–1022, June 2016. 2

[5] Ross Girshick, Ilija Radosavovic, Georgia Gkioxari, Piotr Dollár, and Kaiming He. Detectron. https://github.com/facebookresearch/detectron, 2018. 3, 4

[6] John Paulin Hansen, Alexandre Alapetite, I. Scott MacKenzie, and Emilie Møllenbach. The use of gaze to control drones. In *Proceedings of the Symposium on Eye Tracking Research and Applications*, ETRA '14, pages 27–34, New York, NY, USA, 2014. ACM. 2

[7] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *IEEE Int. Conf. on Computer Vision*, pages 2980–2988, 2017. 2, 3, 4

[8] Eldar Insafutdinov, Leonid Pishchulin, Bjoern Andres, Mykhaylo Andriluka, and Bernt Schiele. Deepercut: A deeper, stronger, and faster multi-person pose estimation model. In *ECCV*, pages 34–50. Springer, 2016. 2

[9] Pascal Knierim, Thomas Kosch, Alexander Achberger, and Markus Funk. Flyables: Exploring 3D interaction spaces for levitating tangibles. In *Prof. of Int. Conf. on Tangible, Embedded, and Embodied Interaction*, pages 329–336. ACM, 2018. 2

[10] Jangwon Lee, Haodan Tan, David Crandall, and Selma Šabanović. Forecasting hand gestures for human-drone interaction. In *Companion of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*, pages 167–168. ACM, 2018. 2

[11] Dushyant Mehta, Srinath Sridhar, Oleksandr Sotnychenko, Helge Rhodin, Mohammad Shafiei, Hans-Peter Seidel, Weipeng Xu, Dan Casas, and Christian Theobalt. Vnect: Real-time 3d human pose estimation with a single rgb camera. *ACM Trans. Graph.*, 36(4):44:1–44:14, July 2017. 2

[12] Sepehr MohaimenianPour and Richard Vaughan. Hands and faces, fast: Mono-camera user detection robust enough to directly control a uav in flight. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 5224–5231, 2018. 2, 4

[13] Pavlo Molchanov, Shalini Gupta, Kihwan Kim, and Jan Kautz. Hand gesture recognition with 3d convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 1–7, 2015. 2

[14] Kai Nickel and Rainer Stiefelhagen. Visual recognition of pointing gestures for human–robot interaction. *Image and Vision Computing*, 25(12):1875–1884, 2007. 2

[15] Mohammad Obaid, Felix Kistler, Gabrielė Kasparavičiūtė, Asim Evren Yantaç, and Morten Fjeld. How would you gesture navigate a drone?: A user-centered approach to control a drone. In *Prof. of 20th Int. Academic Mindtrek Conf.*, pages 113–121. ACM, 2016. 2

[16] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011. 4

[17] Vasil L Popov, Kostadin B Shiev, Andon V Topalov, Nikola G Shakev, and Sevil A Ahmed. Control of the flight of a small quadrotor using gestural interface. In *2016 IEEE 8th International Conference on Intelligent Systems (IS)*, pages 622–628. IEEE, 2016. 2

[18] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017. 2, 4

[19] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018. 4

[20] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. MobileNetv2: Inverted residuals and linear bottlenecks. In *Prof. of IEEE Conf. on Computer Vision and Pattern Recognition*, pages 4510–4520, 2018. 4

[21] Ting Sun, Shengyi Nie, Dit-Yan Yeung, and Shaojie Shen. Gesture-based piloting of an aerial robot using monocular vision. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5913–5920. IEEE, 2017. 2