UNIVERSITÀ
DI TRENTO

Human and animal behavior classification through DKFs

Department of Information Engineering and Computer Science

Project of Distributed Robot Perception

Samuel Bortolin, Alessandro Grassi

Problem Formulation

- Classification between *animal* and *human* classes through the analysis of the movement of such entities inside a Matlab simulation
- Distributed estimation using consensus algorithm using Metropolis-Hastings weights
- The static sensors estimate the position of the entity w.r.t. the global frame

Adopted Models

- GPS with neglectable bias & processing step on thermal camera images in order to obtain estimated distance and direction of the entity
- Bluetooth low energy for sensor communication
- Detection & communication ranges of the sensors

Prediction Step

- Human:

$$p\_human\_pred(k) = A * p\_human(k - 1) + \\ + direction\_human\_predicted * u\_human\_predicted$$

$$P\_human\_pred(k) = A * P\_human(k - 1) * A^t + sigma\_noise\_human^2 * I$$

- Animal:

$$p\_animal\_pred(k) = A * p\_animal(k - 1) + \\ + (direction\_animal\_predicted * u\_animal\_predicted) / scaling\_factor$$

$$P\_animal\_pred(k) = A * P\_animal(k - 1) * A^t + sigma\_noise\_animal^2 * I$$

$$a_i = H_i * R_i^{-1} * z_i$$
$$F_i = H_i^t * R_i^{-1} * H_i$$

Update Step

- Human:

$$P\_human(k) = (P\_human\_pred(k)^{-1} + node\_estimate(k) * F(k))^{-1}$$

$$p\_human(k) = p\_human\_pred(k) + P\_human\_pred(k) * node\_estimate(k) * * (I - node\_estimate(k) * F(k) * P\_human(k)) * (a(k) - F(k) * p\_human\_pred(k))$$

- Animal:

$$P\_animal(k) = (P\_animal\_pred(k)^{-1} + node\_estimate(k) * F(k))^{-1}$$

$$p\_animal(k) = p\_animal\_pred(k) + P\_animal\_pred(k) * node\_estimate(k) * * (I - node\_estimate(k) * F(k) * P\_animal(k)) * (a(k) - F(k) * p\_animal\_pred(k))$$

Classification Step

$$\Lambda\_human(k) = prob(\, z(k) \,|\, model = human,\, z(1), \ldots, z(k-1)\,)$$

$$\Lambda\_animal(k) = prob(\, z(k) \,|\, model = animal,\, z(1), \ldots, z(k-1)\,)$$

$$prob\_human(k) = \frac{\Lambda\_human(k) * prob\_human(k-1)}{\Lambda\_human(k) * prob\_human(k-1) + \Lambda\_animal(k) * prob\_animal(k-1)}$$

$$prob\_animal(k) = \frac{\Lambda\_animal(k) * prob\_animal(k-1)}{\Lambda\_human(k) * prob\_human(k-1) + \Lambda\_animal(k) * prob\_animal(k-1)}$$

The class associated with the highest probability of **fitting the measurements** over time with respect to the pdfs outputted by the two Kalman filters will be the one predicted

Implementation Details

- Simulate the movement of the entity in Matlab generating a random set of coordinates from which the entity will start

- Random direction: in the case of the **human**, it remains **unchanged** for all the simulation, instead for the **animal** it **randomly changes with a certain probability** between one iteration and the other

- The simulation of the entity also includes some **noise on the length of the step**, to do that we randomly lower or raise the norm of the step by a certain value: it starts from a value of *20 meters* and varies with uniform probability in range *[-2.5, 2.5] meters* for the animal model and in range *[-0.05, 0.05] meters* for the human model

Results

With our approach using a default set of hyperparameters:

- The number of sensors/nodes of the network: 50
- Nodes can communicate with other nodes up to a certain distance: 500 meters
- Probability of failure to communicate: 0.001
- Sensors uncertainties: sigma_z_value: 25
- Sensors reveal the entity up to a certain maximum distance: 250
- Probability of not detecting the entity: 0.001
- Human and Animal model uncertainties: 10
- Probability of the animal to change direction: 1/3
- Kalman iterations: iterations = 30
- Minimum number of iterations for computing direction and norm of the control input: 5
- The scaling factor on the step that the animal will take in the previous direction: 3

Results

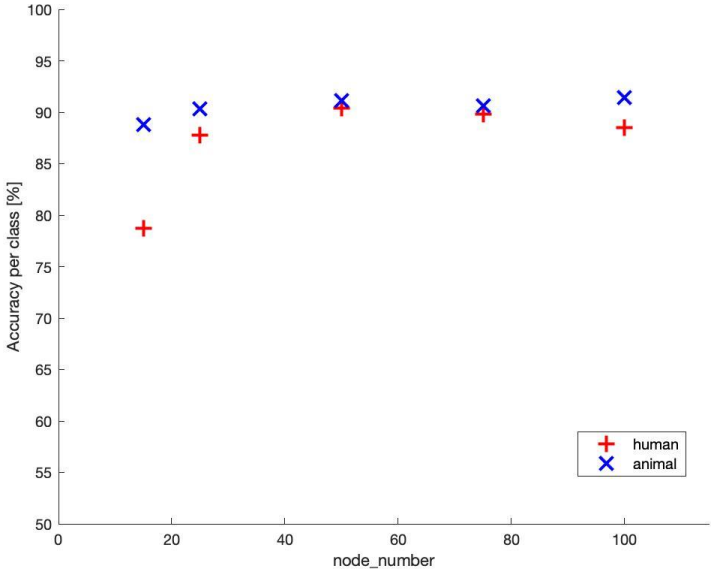
We obtained **90.4%** accuracy over 1000 experiments for each class, **91.1%** accuracy with a mean tracking error of **20.3 meters** on the **human entities** and **97.6%** with a mean tracking error of **13.2 meters** on the **animal entities**

Creating the best scenario with maximum communication and detection capabilities, using low noises for both measurements and the models, changing the following hyperparameters:

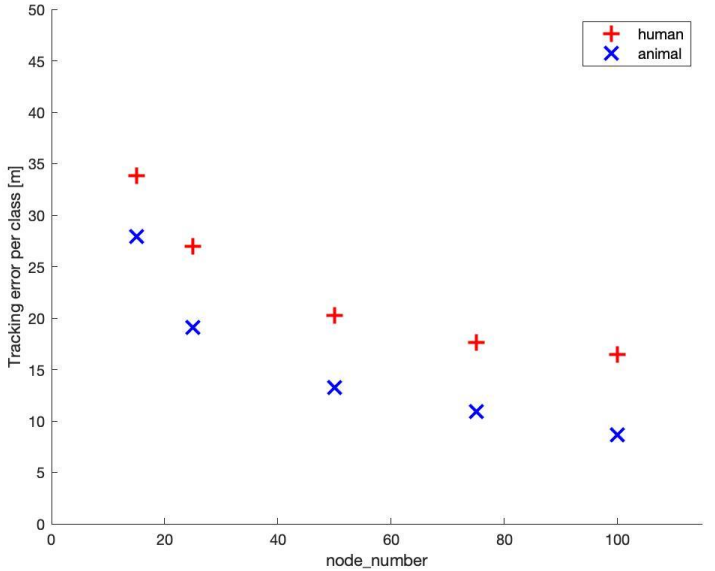
- Nodes can communicate with other nodes up to a certain distance: 500000
- Probability of failure to communicate: 0
- Sensors uncertainties: sigma_z_value: 1
- Human and Animal model uncertainties: 1
- Sensors reveal the entity up to a certain maximum distance: 500000
- Probability of not detecting the entity: 0

We obtained **99.2%** accuracy over 1000 experiments for each class, **100%** accuracy with a mean tracking error of **0.18 meters** on the **human entities** and **98.4%** with a mean tracking error of **0.34 meters** on the **animal entities**

Results

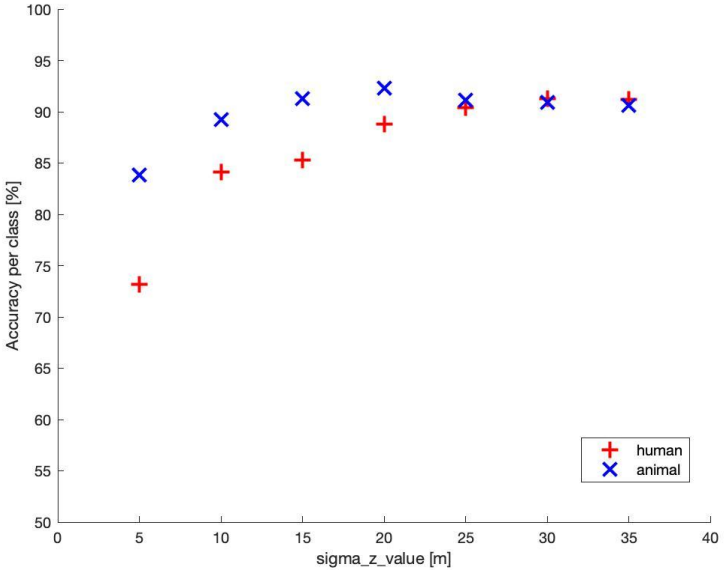


How the True positives (human) and the True negatives (animal) are affected by the number of sensors in the network

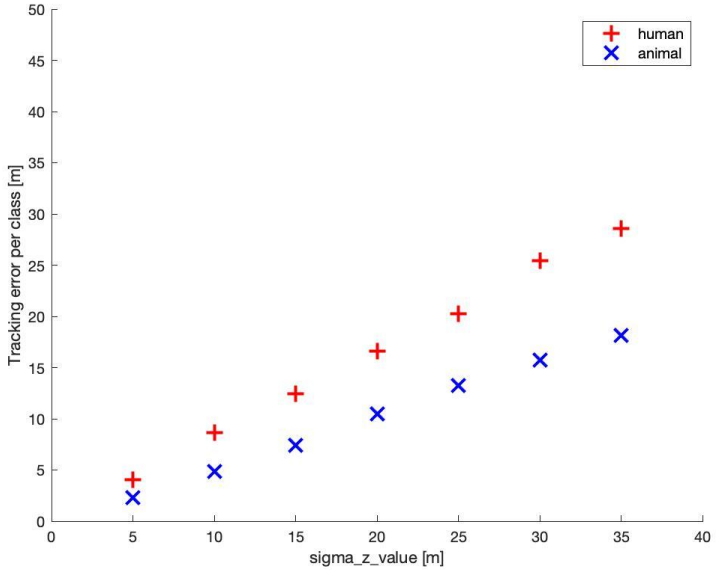


Tracking error with respect to the number of nodes

Results

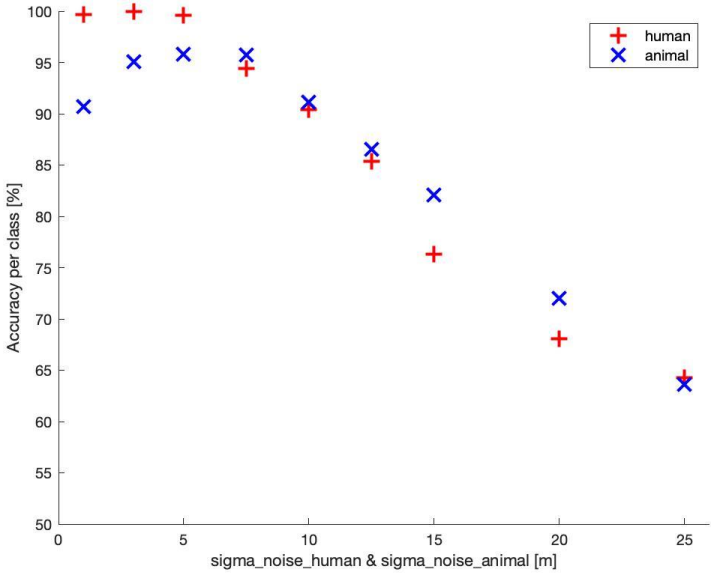


True positive and true negative w.r.t. the variation of sigma_z_value through which the simulation is performed

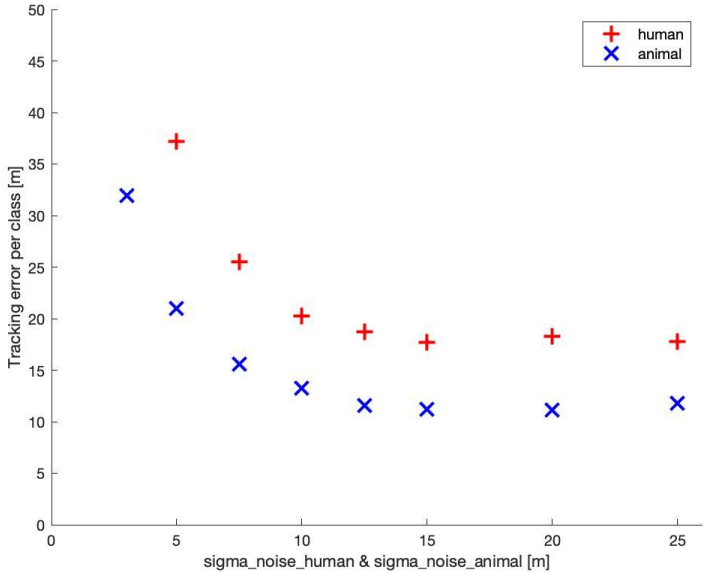


Tracking error with respect to the variation of sigma_z_value

Results



How the True positives (human) and the True negatives (animal) are affected by model noise standard deviations



Tracking error with respect to model noise standard deviations

Conclusions

- Two DKFs have been developed to track two different classes, *human* and *animal*, based on the knowledge of their model
- These DKFs have been used to perform a classification task between the two models, the class associated with the highest probability of **fitting the measurements** over time with respect to the pdfs outputted by the two Kalman filters will be the one predicted
- The resulting algorithm reaches an accuracy of **90.4%** of predicting human entities and **91.1%** of predicting animal entities with the default configuration of hyperparameters
- Testing different hyperparameter configurations, the method has proven to be **resistant to changes in the hyperparameters**, even when noises are increased
- Creating the best scenario with maximum communication and detection capabilities, using low noises for both measurements and the models we obtained **99.2%** accuracy, **100%** accuracy on the **human entities** and **98.4%** on the **animal entities**

Thoughts & Future Work

- Distributed estimation allowed us to obtain better results thanks to the fact that many different sensors in different positions **share their knowledge** about the estimated position of the entity in order to reach a consensus with the others
- **Increased fault tolerance of the network** having nodes that share knowledge with each other with respect to have a centralized node that collects and elaborates information
- This approach also has some downsides, first of all it is necessary to have a **good knowledge of the dynamics of the entities**, it is also important to have a **good range of communication** in order to share with more sensors the own measurements and reach a consensus faster and it is crucial then to have a **good computer vision algorithm** able to output the estimated distance and direction of the entity with respect to the position of the sensor starting from the thermal camera data, with a **good range of detection and with a high precision**
- A possible future direction is to deploy our system on some sensors in a real scenario and test its effectiveness also with real entities moving in the environment under analysis
- This approach is also limited by the fact that it can detect only one entity at the time, as future work it should be possible to track more entities and classify all of them simultaneously

UNIVERSITÀ
DI TRENTO

Human and animal behavior classification through DKFs

Thank you for your attention

Samuel Bortolin, Alessandro Grassi