Comfort-Oriented Social Force Model and Learned Lessons

Miquel Isidre Vidal Carretero, A. Garrell, Luis Garza-Elizondo and R. Alquezar

Abstract— **In this paper, we propose a new Social Force Model (SFM), called COSFM (Comfort-Oriented Social Force Model), that maximizes the comfort of people when robots navigate around them. We describe a navigation system that allows robots to reach their target in small human environments like a flat or small corridors in a safe and comfortable manner. More precisely, the proposed approach is tested with a 3D version of the Social Force Model (SFM) for aerial robots with some restrictions in order to maximize the comfort of nearby humans. To accomplish this commitment, we include a navigation scheme that predicts the humans motion and intention so the robot can safely avoid people. Moreover, to avoid surprises, robots will never go faster than human motion velocity. This contrasts with other conventional works where the environment is neither so tight or close with people and walls. This work is based on previous works of SFM [1] applied to robots that come from an older framework for understanding human movements in crowds [2]. However, we must advise that even if the first results are quite good there is still much work to do to obtain a fully comfortable model. The current model shows great potential to solve difficult decisions, even scale on many elements in close environments.Finally, the results are good enough to think it is an appropriate model for indoor human close environments with applications such as delivering goods or reaching people for help in buildings.**

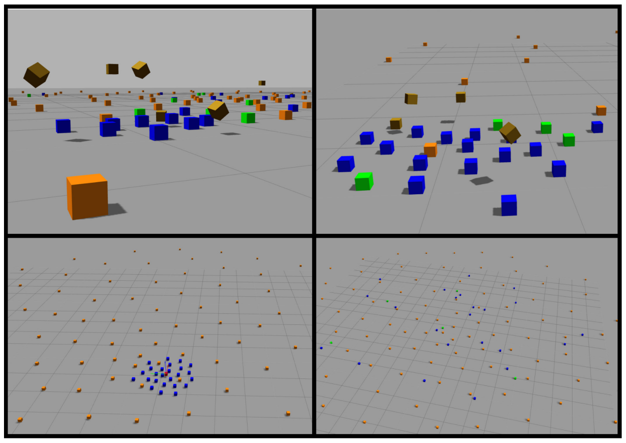
Fig. 1: **SFM simulations with ground and aerial robots.** *Top row:* Blue boxes are people, green boxes are ground robots, gold ones are aerial robots and orange boxes are obstacles. *Bottom row:* Scenarios with more than 200 obstacles, more than 20 people and different robots following people and avoiding collision.

I. Introduction

Currently, it is very common to see an aerial drone flying in a park or a self driving harvester. First, auto-driving cars are working on the road. All these progresses in the field of robotics, self navigation and fast decision making have empowered many industries to reinvent their processes or reach new goals that before were very expensive. For example, from modern underwater robots to self driving cars. In the side of domestic applications, plenty of cleaning robots have spread in many homes. Driving solutions to maintain windows and floors clean. However, real indoor environments present obstacles and highly dynamic objects moving in different directions and velocities, making difficult to achieve a safe navigation within the environment and to reach the final goal with people moving at same time that the robot. The main goal of this article is to propose new ways to face this challenge.

The research has been focused over aerial robots because they are quite common and easy to manage. Deploying on them new navigation algorithms should be simpler work and many homes could handle a drone inside them. Most of aerial robots nearly lack holonomic constraints, which makes their managing easier and more effective. For that reason, we use a drone to achieve our goal.

For this purpose, we push the SFM model one step further looking to robot navigation in human environments without making humans feel unsafe or uncomfortable in close and indoor spaces like an apartment. Many articles have focused on navigation with people [3] either take comfort [4] as a navigation parameter. But as we will see, the time constraint is a hard limit to not pass either prediction on near passerby people make the difference between an aggressive navigation or soft one.

II. Revision of SFM

A. Introduction to SFM

The Social Force Model[2] models movement elements as particles that are attracted by a $\mathbf{F_R^{goal}}$ and suffer/apply repulsion force $\mathbf{F_R^{int}}$ to other elements on the field. Every interaction and final destination defines a sum of existing forces which determines the final force. The model takes into account interaction between robots and people, robots and obstacles and robots and robots.

The total force is defined by the sum of interaction forces from elements, and the attraction forces.

$$\mathbf{F_R} = \mathbf{F_E^{goal}} + \mathbf{F_E^{int}} \tag{1}$$

The interaction force $\mathbf{F_e^{int}}$ is the summation of all the repulsive forces exercised by other pedestrians, objects and robots around him, and is defined as:

$$\mathbf{F_E^{int}} = \sum_{pj \in P} \mathbf{f_{Ej}^{int}} + \sum_{o \in O} \mathbf{f_{Eo}^{int}} + \sum_{r \in R} \mathbf{f_{Er}^{int}} \tag{2}$$

Being $pj \in P$ people around the subject and $o \in O$ the different obstacles and $r \in R$ the robots around it. Obviously this mode could be extended to vehicles, animals or other elements we want to introduce in the equation of forces calculation.

The beauty of this model is that evey force is calculated through a simple function:

$$\mathbf{f_{Ej}^{int}} = A_{Ej} e^{(\frac{d_E - d_{Ej}}{B_{Ej}})} \frac{\mathbf{r_{Ej}(t)}}{d_{Ej}(t)} \tag{3}$$

As E could be any element $\in R$, P, O. Having the following statments d_E,A,B as constant values pending on their element type. On the other hand d_{Ej} as distance between both elements, and the fraction generates the normalized vector pointing from the element to our subject.

B. Ground Robots

In the space of research, in ground robots there are couple of concepts to take in account. First of all, it takes best profit of spaces. Usually, robots need more space or, in our case, we want to take extra space on movement to not bother near humans. In that sense, there is an approximation that takes into account being in the area behind people as less annoying [21], therefore, more comfortable. This approximation adds an extra element to previous SFM calculations in the following form:

$$\mathbf{f_{Ej}^{int}} = A_{Ej} e^{(\frac{d_E - d_{Ej}}{B_{Ej}})} \frac{\mathbf{r_{Ej}(t)}}{d_{Ej}(t)} w(\varphi_{Rj}) \tag{4}$$

Where $w(\varphi_{Ej})$ is described as:

$$w(\varphi_{Ej}) = \lambda_{Ej} + (1 - \lambda_{Ej})(\frac{1 + cos(\varphi_{Ej})}{2}) \tag{5}$$

Where λ_{Ej} defines the strength of the anisotropic, space left on the back, and $cos(\varphi_{iE})$ is calculated as:

$$cos(\varphi_{Ej}) = -\mathbf{n_{Ej}} \cdot \mathbf{e_{pj}} \tag{6}$$

using $\mathbf{n_{Ej}}$, the normalized vector pointing from the robot to p_j, it describes the direction of the force, and e_{pj} is the desired motion direction of the pedestrian *pj* (which is pointing to the goal).

On the other hand, we have the kinodynamic space the robots need to perform their movements. Every step could be a hard calculation and some efforts have been done to find a fast good path for the moving robot [3]. Nevertheless, our tries on a close space with some people has revealed a high cost time of calculation. In short form, it was proposed k random concatenated calculations and chose best space to move, following the SFM model. The bad part is that every step will generate $e \in E$ SFM calculations having a k of 500 it will take too long for our capacity. Without diving deep on this matter and also other approximations, see Fig. 2, has been tried in that way. None of them reveal a fast solution for a local planner.

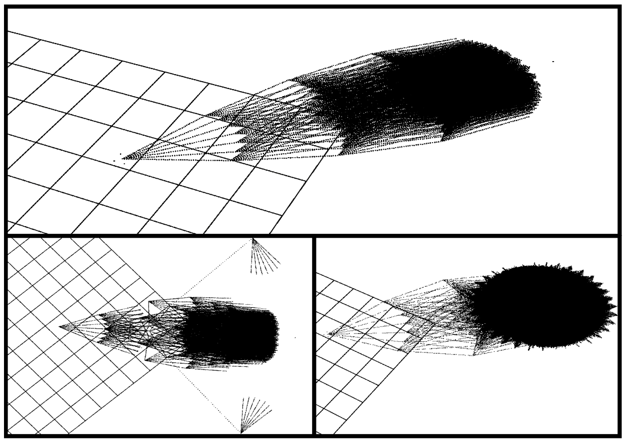
Fig. 2: **SFM Spacial Exploration.** *Top row:* Short range exploration tree affected by SFM. *Bottom row:* Left scenario is a Pruned exploration tree affected by SFM. Right one is next to objective SFM exploration tree.

C. Aerial Robots

Following the previous improvements of ground robots taking into account comfort of the back space left more improvements appear on aerial robots [5]. They propose an improvement to the SFM model that takes into account the space of comfort of people. They propose also the aproximation of taking into account 3D comfort space. Then the SFM model takes following form:

$$\mathbf{f_{Rj}^{int}} = A_{Rj} e^{(\frac{d_R - d_{Rj}}{B_{Rj}})} \frac{\mathbf{r_{Rj}(t)}}{d_{Rj}(t)} \psi(\varphi_{Rj}, \theta_{Rj}) \tag{7}$$

$\psi(\varphi_{Rj}, \theta_{Rj})$ represent the anisotropic factor in a 3D space, which depends on φ_{Rj} , an angle formed between the desired velocity of the pedestrian p_i and the vector $\mathbf{r_{Rj}}$, indicating the distance between the robot and the pedestrian p_j and pointing to him, and θ_{Rj} the angle between the r coordinate plane and z coordinate, formed between the position of the robot and the pedestrian. $\psi(\varphi_{Rj}, \theta_{Rj})$ is defined as:

$$\psi(\varphi_{Rj}, \theta_{Rj}) = w(\varphi_{Rj}) \cos(\theta_{Rj})(h + \xi_{Rj} w(\varphi_{Rj})\eta) \tag{8}$$

where h is the height of the pedestrian, η is a constant defined by us as 1.25, after performing experiments with non-trained volunteers. Further, $w(\varphi_{Rj})$ is described as:

$$w(\varphi_{Rj}) = \lambda_{Rj} + (1 - \lambda_{Rj})(\frac{1 + cos(\varphi_{Rj})}{2}) \tag{9}$$

Where λ_{Rj} defines the strength of the anisotropic, and $cos(\varphi_{iR})$ is calculated as:

$$cos(\varphi_{Rj}) = -\mathbf{n_{Rj}} \cdot \mathbf{e_{pj}} \tag{10}$$

using $\mathbf{n_{Rj}}$, the normalized vector pointing from the robot to p_j, it describes the direction of the force, and e_{pj} is the desired motion direction of the pedestrian *pj* (which is pointing to the goal).

The drone anisotropic factor is different than the human one, due to the fact that the drone is in the air all the time,

Interaction	k	A	B	d	λ
Per-Per [20]	2	1.25	0.1	0.2	0.5
Per-Per [21]	4.9	10	0.34	0.16	1
Robot-Person [22]	2.3	2.66	0.79	0.4	0.59
Drone-Person	4.45	3.35	0.565	0.295	0.55

TABLE I: **Aerial Social Force Model Parameters.** Parameters learned for Robot-drone interaction after applying the minimization process.

it does not have the same limitations as the human, so the anisotropic factor is defined by:

$$\psi(\varphi_{Rj}, \theta_{Rj}) = w(\varphi_{Rj}) \cos(\theta_{Rj}) \tag{11}$$

From the previous and current sections we get the parameters [5] on Table:I.

D. Prediction

For our model we also include a simple prediction system to estimate future velocity of moving elements, in our case humans. As we will see next sections our comfortable oriented SFM sets maximum velocity of the drone to 0.8 max velocity of near humans. This increases the safety perception of near people but at the same time raises a challenge to avoid their path when the robot will move slower than them.

For this exercise we use a really simplistic approach. Every time it notices a new position from someone the predictor engine records their location and time stamp. After some records the predictor has a vector of positions and a vector of times:

$$X = (x_1, x_2, ..., x_{11}) \tag{12}$$

$$T = (t_1, t_2, ..., t_{11}) \tag{13}$$

Then, we can elaborate a vector of estimated velocities:

$$v_n = \frac{x_n - x_{n-1}}{t_n - t_{n-1}} = \frac{\Delta x_n}{\Delta t_n} \tag{14}$$

With final result:

$$V = (v_1, v_2, ..., v_{10}) \tag{15}$$

For our exercise we record the last 11 positions getting the last 10 approximated velocities.

To predict people's future velocity:

$$v'_{n+1} = \frac{\sum_{j=1}^{n} v_n}{n} \tag{16}$$

III. Simulation Environment

A. Introduction to environment

First of all we have create a simple environment, using Gazebo Ros, to check if the model can handle hundreds of static elements, tens of people and robots following people either navigating on it without collision and consuming low resources. Fig. 1. All simulations done in this article have been done in a desktop computer with 6Gb Ram with double Xeon E5520 processors.

We have worked on a development on a local planner base on SFM. This version takes into account some parameters to increment the comfort of near people. The concept of local planner is common in many frameworks[12][4], such as ROS. The main idea is the existence of a Global Planner that carries with it the responsibility to trace a global path to reach a target and a local planner that will drive the robot through the changing reality that can need time for large or accurate calculations for replanning the proposed path. For our final exercise we use a constant calculated global planner. That engine always delivers the same route to the robot. That way we can repeat the exercise and compare the different behaviors of our local planner in near equal conditions. Global planners should take care to solve situations that only could be predicted or solved with bird sight. For example SFM models suffer from the same issue that potential fields navigation algorithms with the local minima[7]. When this happens the global planner should take action to find a path avoiding these situations.

For our commitment we have started working with Navigation ROS framework, also similar to other options but at the end reduced to a simpler option. As we will check some issues appear on traditional frameworks over 3D aerial robots. Both frameworks take in account the option on Global Planner and Local Planner. In fact we reproduced [3] other based SFM local planners to improve their result, left square Fig. 10.

B. Configuration of near real environment

One of the problems to manage on simulations is to generate a virtual environment as realistic as could be. In order to achieve this commitment we have used Gazebo Ros as simulation environment. Gazebo is currently a separated software from ROS for managing simulations, but in our version of environments (ROS Indigo) was still a package of ros.

Create a realistic environment began with reproducing a real scenario based on a real flat (Fig. 3). We take the floor plans of a team member. Modeled as a simpler version (squared it) and raised it on the floor builder integrated on Gazebo. After that we had a realistic physical environment where our experiment could be placed.

Second element to develop was generating a group of virtual actors that could walk around the flat. Each of them was modeled as person following the primal version of SFM defined by Helbing [2]. Their behavior follows previous SFM model. As we want to launch a large range of experiments to observe their results. We defined 6 different human profiles, having each of them a different route inside the previous flat. We can observe their planned path on the small six right squares on the Fig. 10. We want to call special attention on paths defined on the upper left and bottom right. Both are designed to have frontal collision with our virtual drone. The first path will approach the drone to the upper left room and the second will try to push the drone against the wall on the small corridor.

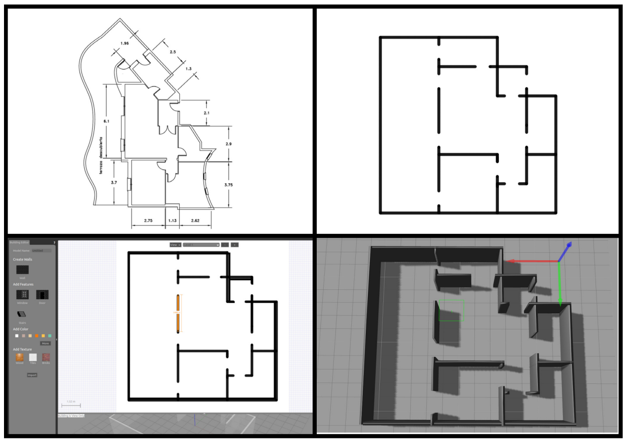

Fig. 3: **Building a realistic map.** *Top row:* Transform a real map to easy to reproduce one. *Bottom row:* Build the map and final result in the 3D simulator.

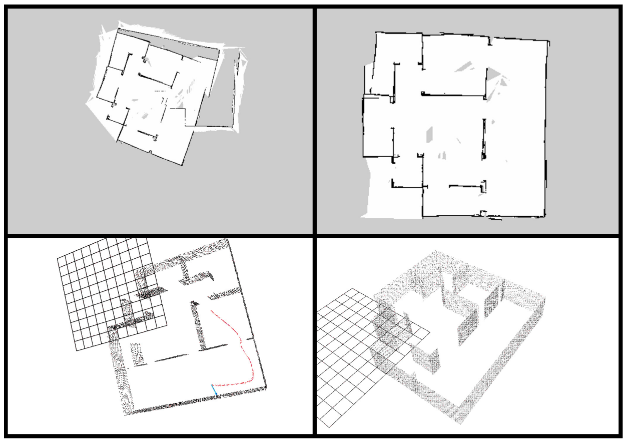

Fig. 4: **Reproducing Real Experiment.** *Top row:* Two explored maps inside the simulator. *Bottom row:* Odometry representation of walls and path of the drone.

Our experiment will launch at least 50 runs on 8 scenarios. First scenario will be with no people and our drone guided only with the global planner. Second scenario is like the previous one with our local planner. Third scenario will bring the first virtual actor into the simulation. The following scenarios will add one extra virtual actor until reaching 6 virtual actors on the simulation. The final 2 scenarios will generate the previous collision situations explicitly provoked.

Finally we have been using a virtual Drone that emulates the real behavior of a commercial AR Drone 2.0 [6]. In our preliminary experiments we have already noticed their similarity with real aerial robots. It suffers drift either inertial and does not answer instantly to commands. As previous virtual actors, get direct actions from our SFM model engine for our virtual Drone we go through full stack simulation.

C. Learned Lessons

The presented documents want to focus in our model (COSFM), however, we would like to make a small stop to share some experiences on working with the local path planner, small environments, compatibility of 3D environments and current existing planners. Finally, a special chapter for doors.

1) Oscillations: Many navigation systems suffer from the same challenge. When they want to reach a point in space, it could happen, that your inertia, either slow answer on actuators or other elements makes your robots miss it out. When they come back to reach it could easily start a loop rounding around that target. To reduce this situation we have deployed a couple of mechanisms. First of all as soon as it is near the next step of its route, just accept as matched and ask for the following point of the route to the global planner. Secondly and more elegant we create a $n+1$ dimension for managing space in our world. For example a 2D world will work with 3D vector coordinates. This third coordinate just acts as a reducer for our SFM engine. As F_{goal} pushes our Drone to the coordinates of the goal, if it has a $n+1$ coordinate that could not be reached by their movement, but it is taken into account during their SFM calculations as soon as the Drone gets near that coordinate it gets a bigger force portion until one time it gets most of F_{goal} generated.

Drawbacks appear on both workarounds. On the case of accepting the near point as matched on the route could bring difficult routes when a turnover 90 square appears on the route or a door has its space for the next step, that margin should be reduced to a minimum. In the other hand the $n+1$ dimension solution could make everything run a little bit slower as for each step on the route some $force$ is derived to $nowhere$ place.

2) Walls: One of the big challenges on managing low cost time calculations over the local planner is managing the information provided by the sensors from the walls. Usually that data is managed as a cloud of points and the map engine will interpret it as a wall. This works correctly for the global planner. The problem begins with a SFM based local planner. As soon as we work with massive cloud points sourced by the small corridors or walls on the flat our SFM will deliver slow answers to the local planner and perhaps it reaches late. We simplified it a little bit for the walls. As soon as we know a 3D cloud of points belongs to a wall, we manage it as a line of points at the same level that our drone. As we can see on Fig.4 bottom row the full knowledge of the cloud points could be a challenge for the SFM engine. Even when we use the near restriction it will still handle a large data set for no real profit.

3) Current Local Planners: In order to be able to make better comparison of our proposal we face the problem that not all local planners are prepared for the 3D environment. This could seem an obvious reality, but it wasn't so evident on some experiments.

4) Doors: As we run our experiments in a simulation based on a real flat. Our doors are a small space to go thought. Going through a door could generate many oscillations, trying to not collision with one side of the door pushes to the other side and begins dancing between them. Also doors are so close spaces that is really difficult if one of our virtual actors tries to go across it at the same moment our drone tries it, it doesn't have space to make any evasive maneuver. At one point we realized that this problem also happens with real humans and also when people try to drive their car in a small space it also generates this kind of behaviors. Doors made us tune more accurately COSFM than any other element in the stage (even the virtual actors). We reduced a little bit our parameters for SFM and let it drive carefully through them.

IV. COSFM

A. Re-Simplication

As we have described in previous sections SFM has many improvements to make it more accurate their $forces$ over the robot, pending on 3D position or angle front facing the person. In our case we finally have chosen the simpler form. One that doesn't take into account either $\psi(\varphi_{Rj}, \theta_{Rj})$. Mostly it has been discarded due to a double fact. First we are simulating indoor flight, that means that the improvement of the Areal SFM could easily hit the roof. Secondly for our simulation the maximum population we will manage is 6 virtual actors, which reduces in great manner the actual benefit of this improvement.

B. Focus

COSFM focus on getting the base for a local planner that targets maximize the comfort of near people based in following directives:

1) Velocity: The drone velocity could be one of the bigger bothering factors to near people[4]. Then we limit the maximum velocity of the drone to 0.8 velocity of near people. This parameter becomes dynamic and with a minimum value. If all people are stopped let it be 2.8 km/h which is slower than common human walk speed 4 km/h.

2) Close Spaces Oriented: Our model is oriented to closed spaces. Perhaps in the future research could be oriented to open spaces, but the aim of this model is to maximize comfort and this concept is strongly linked to short range space. At the moment it has much less to bring in open spaces in comparison of closed ones.

3) Human collision as last option: As we introduced in previous chapters, some executions put our virtual drone in difficult situations where our virtual actor will push it to the wall or close it in a small room. Our model will always try to avoid collision with people, but in case of conflict between people and wall, it will choose wall. This happens automatically with SFM parameters. Nevertheless this should be taken in account when working on really close spaces.

C. Timeless Prediction

Our improvement on SFM navigation is based on this "Timeless Prediction". As we can observe on Fig. 5 our Drone would be attracted by the goal, but the main challenge will be to not collide with anyone. On the top left we can see a couple of possible paths the drone could take if all people would be static. If we bring some movement on the scenario (other 3 quarters) we can easily observe that one or the other path are better depending on people decisions. It

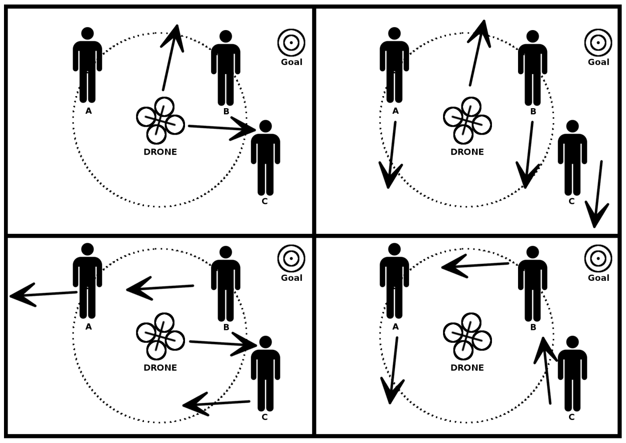

Fig. 5: **COSFM.** *Top Left:* On a static stage a couple paths has clear ways. *Top Right:* If all people move down then the upper path is the best option.*Bottom Left:* If all people move left then the right path is the best option. *Bottom Right:* But if all people are moving perhaps it's better not move.

could also be that the best option is to remain static (bottom right quarter).

From works [14][23][3] we can observe many ways to manage when/where near mobile elements will be and where our navigated element could go. But if we are thinking about low time cost calculations or look for a low iteration model soon we will find that developing any model over time could be difficult to manage. At the same time if we observe again Fig. 5 we will realize that it is not mandatory for a short period to see the time as variable. We can use all the future projections $f_j \in F$ at the same moment to reveal the proper path:

$$\mathbf{F_R^{int}} = \sum_{pj \in P} \mathbf{f_{Rj}^{int}} + \sum_{f_j \in F} \mathbf{f_{Rj}^{int}} + \sum_{o \in O} \mathbf{f_{Ro}^{int}} \tag{17}$$

For every moving element, in our case virtual actors, we create a virtual representation of them in the direction of their predicted velocity for 1 second further. This gives a small line of projections of people that will reject movements of our Drone. While it is true that this will increase our calculations per round at the same time we are reducing iterations over time to predict future moves and transforming to the SFM the intention of mobile elements in form of force elements. In the SFM framework this becomes an easy projection of current moving elements to the predicted direction.

D. Math Model

This chapter focuses on giving the whole math model in one chapter, first we have an f_E^{goal} for each moving element in our environment, in our case virtual actors and our aerial robot(Drone). This f_E^{goal} attracts the element to their next target position and could be defined as:

$$\mathbf{f_E^{goal}} = k(\mathbf{v_E^0} - \mathbf{v_E}) \tag{18}$$

Being v_E^0 our desired velocity (from our subject to their target) and v_E our current velocity. This autoregulates the attraction force until it gets the desired velocity. k is a constant defined in previous chapters (see Table I).

Our virtual actors are modeled like Helbing [2] and only suffer repulsion force from other elements and attraction force from their f_P^{goal}. Their behavior is defined as:

$$\mathbf{F_P^{int}} = f_P^{goal} + \sum_{e_j \in E} \mathbf{f_{Pj}^{int}} \tag{19}$$

Where f_{Pj}^{int}

$$\mathbf{f_{Pj}^{int}} = A_{Pj} e^{(\frac{d_P - d_{Pj}}{B_{Pj}})} \frac{\mathbf{r_{Pj}(t)}}{d_{Pj}(t)} \tag{20}$$

Like previous step, A, B, k, d could be found at Table I.

At this point we have virtual actors that behave as humans and target their local goals. For modeling our aerial robot we apply the same basic SFM model:

$$\mathbf{F_R^{int}} = f_R^{goal} + \sum_{e_j \in E} \mathbf{f_{Rj}^{int}} \tag{21}$$

Where f_{Rj}^{int}

$$\mathbf{f_{Rj}^{int}} = A_{Rj} e^{(\frac{d_R - d_{Rj}}{B_{Rj}})} \frac{\mathbf{r_{Rj}(t)}}{d_{Rj}(t)} \tag{22}$$

and we enhance it with the prediction having:

$$\mathbf{F_R^{int}} = f_R^{goal} + \sum_{e_j \in E} \mathbf{f_{Rj}^{int}} + \sum_{f_j \in F} \mathbf{f_{Rj}^{int}} \tag{23}$$

as $f_j \in F$ are the predicted positions of the mobile elements $\forall positions$ between current position and predicted one we generated elements $f_e = \{e_{position}......e_{position} v' \Delta time\}$. For our actual experiment we use and increment of 0.01 second per projection.

Current filters have been applied to previous model to enhance comfort:

1) Altitude: For any F_R^{int} where z component suffers a modification until it reaches a 0.5m altitude. That makes the drone fly at 0.5m over the floor and only for avoiding obstacles it goes to a different altitude.

2) Velocity: For $min(v_e \in V)$ where v of the elements in the scenario, our aerial drone sets their Max Velocity to $0.8 v_{min}$.

V. Experiments

A. Scenario

The current experiment is developed inside Gazebo simulator. The virtual world will have a simulated flat (Fig. 3) with a simulated drone[6] that will follow a constant path delivered by the global planner constantly. This global planner will deliver the pathss shown on Fig.10. As we mention on previous chapters a couple of paths are designed to collision with the done in the corridor or the L room.

We defined 8 different scenarios, one per number of people, one without people and one without local planner, only global planner.

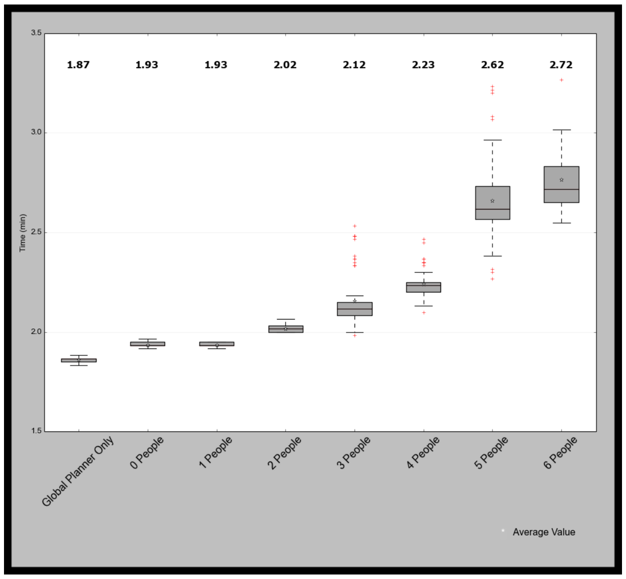

Fig. 6: **Route Time**: How long does the drone take to run whole path in each scenario. Our COSFM takes a bit longer, and gets longer as more people walk on the scenario.

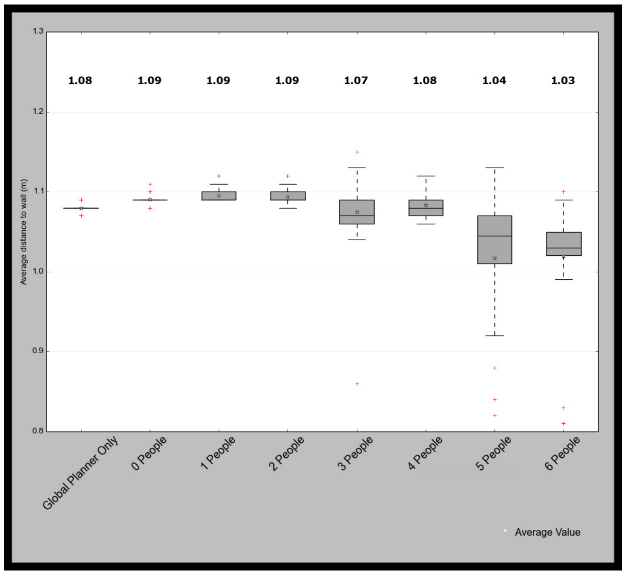

Fig. 7: **Wall Distance**: Average distance of the drone to the wall. More is better, as we can observe that until we reach 4 people it doesn't really affect it.

B. Execution

The execution of this experiment has been developed with an automatic mechanism that launches the scenario continuously, saves the ROS Bag message for each run. This has generated more than 850 ROS Bags, after eliminating the corrupt bags and unlaunched simulations we get a total of 604 bags. With a minimum of 53 bags of 6 people experiment to a maximum of 101 without people or local planner we get a wide range of data to make the analysis of our COSFM in simulation.

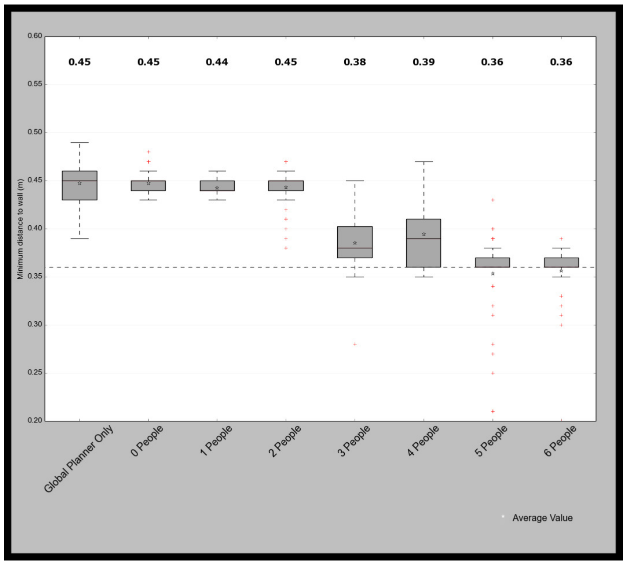

Fig. 8: **Proximity of the wall**: The minimum distance to the wall, as we can see with 4 or more people, the drone begins to collide with the wall in some simulations.

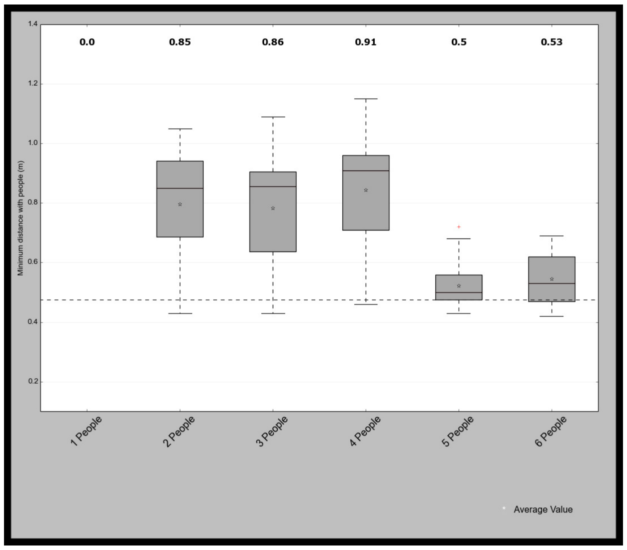

Fig. 9: **Proximity of people.**: Generally, our method doesn't collide with people.

C. Data analysis

From previous simulations, we have raised more than 10Gb of data messages exclusively for COSFM. To process that amount of data we have developed a range of scripts to process their content and apply some logic. From data messages we have selected the following Key Performance Indicators (KPI) per run:

1) Number of people: : Number of people rounding on that simulation launch.

2) Time to route: : The amount of time the virtual drone need to accomplish the whole route. That means enter the flat to reach again the small hall. This time has been measured on simulation time scale, not the real scale. As the simulator runs morre slowly than real time, this could be a problem.

3) Average distance of the wall: : After recording all the position of the virtual drone, how far was it from the wall in average.

4) Minimum distance of the wall: : How near it has been to the walls. This parameter will reveal collision with the wall.

5) Minimum distance of people: : How near it has been to the people. This parameter will reveal a collision with someone.

VI. Conclusions

After observing the COSFM simulations hundreds of times, and the data analysis, we clearly see one good and one bad. Generally our local planner tends to avoid collisions and doesn't bother people. Many times that a couple of virtual actors overun our drone, it reaches a solution. But it could not hide the reality that sometimes in our simulations we suffer some colisions between the drone and our virtual actors. Also we should find a better solution than making the aerial robot collide with the wall when it has to chose between wall or virtual actor. In fact perhaps we should better land (and let the drone be stepped on), as it is less dangerous than hitting the wall and bouncing against the virtual actor.

If we check the figure 6 we can see that respect a no local planner it doesn't bring so much overhead, it makes the route nearly in the same time. Also if we check the figure 7 we can observe it tends to let more space between the drone and the wall.

Over the collision space we would work on figures 8 and 9. Both have a slashed line showing the colission ditance. As we can appreciate for people it is a bit larger. In both scenarios we can appreciate that it gets worst with the provoked collision in small spaces. Either with 5 or 6 people some simulations our drone hits the wall or, what's worst, hits someone.

Also it has been shown over many simulations that prediction is a key point over COSFM, as being slower than people always will be a challenge to avoid the colision, we should anticipate it in a more clever way. This could be a deep field for research.

Finally we understand there is work to do, but and same time we are really pleased for the shown results. It is a really tight space, with small doors to cross or many people moving for the size of the flat. Generally our COSFM planner has worked great and shown as many not so obvious solutions for solving navigations challenges.

VII. Future Works

When we review the videos of the different navigation experiments, we feel that perhaps even humans couldn't drive better neither would be resolved better by other local-planner methods.

Future works will be focused on fine tuning the current method, trying to avoid collisions. Get some tests with the same scenario with human drone pilots, also check the response of other planners in these scenarios.

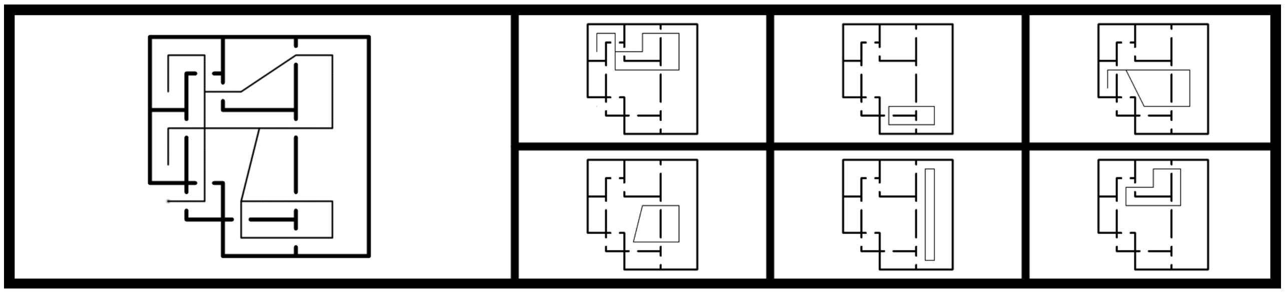

Fig. 10: **Paths.** *Left:* Full path driven by the drone. *Right 6 boxes:* This are the 6 paths followed by the 6 people on the flat.

Finally, we plan to perform real-life experiments. The scenario has been based in an existent flat where we could try if in a real scenario it works properly.

ACKNOWLEDGMENT

This work couldn't be done without the support and patience of my beloved wife Marta, neither the support of A.Garrell nor L. Garza who guides and tutorizes me. The authors are with Institut de Robotica i Informatica Industrial (CSIC-UPC). Llorens Artigas 4-6, 08028 Barcelona, Spain. miquel@miquelvidal.com and {lgarza, agarrell, fherrero, ralqueza}@iri.upc.edu .

REFERENCES

[1] A. Garrell, and A. Sanfeliu. Cooperative social robots to accompany groups of people. The International Journal of Robotics Research 31.13 (2012): 1675-1701.

[2] D. Helbing, and P. Molnar. Social force model for pedestrian dynamics. Physical review E 51.5 (1995): 4282.

[3] G. Ferrer, and A.Sanfeliu. Proactive Kinodynamic Planning using the Extended Social Force Model and Human Motion Predition in Urban Environment. 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems.

[4] E. A. Sisbot and L. F. Marin-Urias and R. Alami and T. Simeon. A Human Aware Mobile Robot Motion Planner. 2007 IEEE Transactions on Robotics Journal: 874-883.

[5] Luis Alberto GARZA ELIZONDO. Social-aware Drone Navigation using Social Force Model. Master Thesis, October 2016.

[6] H.Huang and J.Sturm. Tum Simulator. Ros package at `http://wiki.ros.org/tum_simulator`, May 2014.

[7] Da Silva, Rafael Rodrigues, et al. SafeGuardPF: Safety Guaranteed Reactive Potential Fields for Mobile Robots in Unknown and Dynamic Environments. arXiv preprint arXiv:1609.07006, 2016

[8] D.M. Wilkes, R. T. Pack, A. Alford, and K. Kawamura. Hudl, a design philosophy for socially intelligent service robots. In American Association for Artificial Intelligence. 1997.

[9] H. Ishiguro, T. Ono, M. Imai, T. Maeda, T. Kanda, and R. Nakatsu. Robovie: an interactive humanoid robot. Industrial robot: An international journal 28, no. 6 (2001): 498-504.

[10] K.L. Koay, A. Sisbot, D. S. Syrdal, M. L. Walters, K. Dautenhahn, and R. Alami. Exploratory Study of a Robot Approaching a Person in the Context of Handing Over an Object. In AAAI spring symposium: multidisciplinary collaboration for socially assistive robotics, pp. 18-24. 2007.

[11] K. Dautenhahn, M. Walters, S. Woods, K. L. Koay, C. L. Nehaniv, A. Sisbot, R. Alami, and T. Simon. How may I serve you?: a robot companion approaching a seated person in a helping context. In Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction, pp. 172-179. ACM, 2006.

[12] M. Habbecke, and L. Kobbelt. Automatic registration of oblique aerial images with cadastral maps. European Conference on Computer Vision. Springer Berlin Heidelberg, 2010.

[13] W. Van Toll, and R. Geraerts. Dynamically Pruned A* for re-planning in navigation meshes. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2015.

[14] C-H. Lin, and K-T. Song. "Robust ground plane region detection using multiple visual cues for obstacle avoidance of a mobile robot." Robotica 33.02, 2015.

[15] D. He, H. Ren, W. Hua, G. Pan, S. Li, and Z. Wu. FlyingBuddy: augment human mobility and perceptibility. Proceedings of the 13th international conference on Ubiquitous computing, pp. 615-616. ACM, 2011.

[16] S. Schneegass, et al. Midair displays: Concept and first experiences with free-floating pervasive displays. Proceedings of The International Symposium on Pervasive Displays. ACM, 2014.

[17] J. Nagi, et al. Human control of UAVs using face pose estimates and hand gestures. Proceedings of the ACM/IEEE international conference on Human-robot interaction. ACM, 2014.

[18] D. Arroyo, C. Lucho, S. Roncal and F. Cuellar. Daedalus: a sUAV for human-robot interaction. In Proceedings of the ACM/IEEE international conference on Human-robot interaction, pp. 116-117. ACM, 2014.

[19] E.T. Hall. The hidden dimension, 1966.

[20] M. Luber, J. Stork, G. D. Tipaldi, and K. O. Arras. People tracking with human motion predictions from social forces. In Proceedings of the IEEE International Conference on Robotics and Automation , pp. 464-469, 2010.

[21] F. Zanlungo, T. Ikeda, and T. Kanda. Social force model with explicit collision prediction. EPL (Europhysics Letters), 2011.

[22] G. Ferrer, A. Garrell, F. Herrero and A. Sanfeliu. Robot social-aware navigation framework to accompany people walking side-by-side. Autonomous Robots, 2016.

[23] L. Zhu, X. Cheng, and F. G. Yuan. A 3D collision avoidance strategy for UAV with physical constraints. In: Measurement: Journal of the International Measurement Confederation 77 (2016), pp. 4049. 06.