CZECH TECHNICAL UNIVERSITY IN PRAGUE

Faculty of Electrical Engineering

Bachelor thesis

Jan Langr

Odor source localization using swarm of unmanned helicopters

Department of Cybernetics

Thesis supervisor: **Ing. Martin Saska, Dr. rer. nat.**

Prohlášení autora práce

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne 24. 5. 2013 .....

Declaration

I hereby declare that I have completed this thesis independently and that I have used only the sources (literature, software, etc.) listed in the enclosed bibliography.

In Prague on 24.5.2013 .....

České vysoké učení technické v Praze
Fakulta elektrotechnická

Katedra kybernetiky

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Student: Jan Langr

Studijní program: Kybernetika a robotika (bakalářský)

Obor: Robotika

Název tématu: Lokalizace zdroje pachu pomocí roje bezpilotních helikoptér

Pokyny pro vypracování:

Cílem práce je navrhnout a implementovat algoritmus pro řízení roje bezpilotních helikoptér v úloze sledování mraku kouře (pachu) s cílem nalezení jeho zdroje. Navržený algoritmus bude založený na metodě Particle Swarm Optimization (PSO) [2]. V implementovaném přístupu budou jednotlivé částice PSO reprezentovat fyzické roboty, které se budou pohybovat na základě pravidel PSO. V rámci práce bude do těchto pravidel integrován kinematický model helikoptér a omezení dané fyzickou realizací PSO částic (bezkolizní pohyb). Student bude dále studovat vliv omezení daných relativní lokalizací entit robotického roje.

1. Integrovat kinematický model helikoptéry zahrnující reálná omezení jejího pohybu [1].
2. Implementovat jednoduchý model pohybu mraku kouře pro potřeby testování algoritmu a jeho vizualizaci.
3. Navrhnout pro danou aplikaci vhodnou ohodnocovací funkci PSO.
4. Integrovat omezení daná relativní lokalizací členů roje a jejich reálnou velikostí (kolize).
5. Ověřit funkčnost metody s využitím dostupné robotické platformy. V průběhu řešení projektu vedoucí práce rozhodne, zda navržená metoda bude otestována krátkým experimentem s 2-3 helikoptérami, na lanovém multi-robotu, případně na systému SyRoTek [3]

Seznam odborné literatury:

[1] Taeyoung, L.; Leoky, M.; McClamroch, N.H.: Geometric tracking control of a quadrotor UAV on SE(3). Decision and Control (CDC), 49th IEEE Conference on, pp.5420-5425, 15-17 Dec. 2010.

[2] Kennedy,J. and Eberhart,R.C.: Particle swarm optimization. In Proceedings International Conference on Neural Networks IEEE, volume 4, pp. 1942–1948, 1995.

[3] Kulich, M. - Košnar, K. - Chudoba, J. - Faigl, J. - Přeučil, L.: On a Mobile Robotics E-learning System. In Proceedings of the Twentieth European Meeting on Cybernetics and Systems Research. Vienna: Austrian Society for Cybernetics Studies, 2010, p. 597-602. ISBN 978-3-85206-178-8.

Vedoucí bakalářské práce: Ing. Martin Saska, Dr. rer. nat.

Platnost zadání: do konce zimního semestru 2013/2014

prof. Ing. Vladimír Mařík, DrSc.
vedoucí katedry

L.S.

prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 10. 1. 2013

Czech Technical University in Prague
Faculty of Electrical Engineering

Department of Cybernetics

BACHELOR PROJECT ASSIGNMENT

Student: Jan Langr

Study programme: Cybernetics and Robotics

Specialisation: Robotics

Title of Bachelor Project: Odor Source Localization Using Swarm of Unmanned Helicopters

Guidelines:

The goal of this work is to design and implement an algorithm for control of swarms of unmanned helicopters in the task of odor source localization. The proposed algorithm is based on the Particle Swarm Optimization (PSO) [2] technique. In the implemented approach, individual particles of PSO represent physical robots that will move under the rules of PSO. Into these rules, a kinematic model of helicopters and limits given by the physical implementation of PSO particles (collision-free motion) will be integrated. Student will also study the influence of constraints of the relative localization between swarm particles.

1. Integrate the kinematic model of helicopters with included motion constrains [1].
2. Implement a simple model of smoke cloud movement for testing the algorithm and its visualization.
3. Design a PSO fitness function suitable for the given application.
4. Integrate constraints of the relative localization between swarm members and limits given by their real size.
5. Verify the functionality of the method using the available robotic platforms. During the project solution, thesis supervisor will decide whether the proposed method will be tested by a simple experiment with 2-3 helicopters, using a rope multi-robot system or with SyRoTek platform [3].

Bibliography/Sources:

[1] Taeyoung, L.; Leoky, M.; McClamroch, N.H.: Geometric tracking control of a quadrotor UAV on SE(3). Decision and Control (CDC), 49th IEEE Conference on, pp.5420-5425, 15-17 Dec. 2010.
[2] Kennedy, J. and Eberhart, R.C.: Particle swarm optimization. In Proceedings International Conference on Neural Networks IEEE, volume 4, pp. 1942–1948, 1995.
[3] Kulich, M. - Košnar, K. - Chudoba, J. - Faigl, J. - Přeučil, L.: On a Mobile Robotics E-learning System. In Proceedings of the Twentieth European Meeting on Cybernetics and Systems Research. Vienna: Austrian Society for Cybernetics Studies, 2010, p. 597-602. ISBN 978-3-85206-178-8.

Bachelor Project Supervisor: Ing. Martin Saska, Dr. rer. nat.

Valid until: the end of the winter semester of academic year 2013/2014

L.S.

prof. Ing. Vladimír Mařík, DrSc.
Head of Department

prof. Ing. Pavel Ripka, CSc.
Dean

Prague, January 10, 2013

Acknowledgement

I would like to thank Dr. Martin Saska for his help and useful tips throughout this work.

Abstrakt

Cílem této práce je modifikovat základní PSO algoritmus pro účel nalezení zdroje kouře pomocí roje bezpilotních helikoptér. Jelikož PSO algoritmus vyžaduje, aby členové roje znali svou pozici v prostoru, musí být vybráno a využito několik možností absolutní lokalizace. V zájmu zabránění kolizím členů roje musí také být použit prostředek relativní lokalizace robotů. Nejprve bude implementována základní verze PSO algoritmu s jednoduchou hodnoticí funkcí bez uvážení jakýchkoliv omezení. Po naladění parametrů algoritmu bude aplikován na nalezení zdroje kouře a dále testován. Poté budou implementovány modifikace algoritmu zohledňující omezení reálných robotů (kvadrukoptér) a algoritmus bude opět testován s jednoduchou hodnoticí funkcí. Ve výsledku budou veškeré modifikace spojené do jednoho algoritmu, který bude zaměřený na lokalizaci zdroje kouře a zároveň bude zohledňovat omezení daná reálnými roboty.

Abstract

The goal of this thesis is to modify the Particle Swarm Optimization algorithm to allow its use with small UAVs and utilize it for the task of locating the source of a smoke. Since the PSO algorithm requires the swarm individuals to know their immediate position, several means of absolute localization of the robots have to be devised and used. In order to avoid collisions, a means of relative robot localization must be used. First, a simple version of the PSO algorithm using particles with no constraints will be used in combination with a simple fitness function. After tuning the algorithm's parameters, it will be used to find a source of a smoke and further tested. Then, the PSO algorithm modified using the real quadrotor constraints will be tested with the same basic fitness function. Eventually, the combination of these two apporaches will yield an algorithm applicable on real quadrotors and the task of locating a smoke source.

Contents

List of figures

List of tables

List of algorithms

1 Introduction

One of the aims of mobile robotics and artificial intelligence is to substitute a human being in performing certain tasks. This may be especially useful if the performed task presents a health risk. For instance, locating and neutralizing a gas leakage that could at any moment cause an explosion or lead to suffocation or intoxication of people. The aim of this work is to study whether the use of flying automated robots behaving according to the rules of a modified PSO algorithm can be used to perform this precise task.

In this work, several separate steps will be followed in order to achieve the given goal. At first, a simple version of the PSO algorithm with an easy-to-visualize fitness function will be implemented. This will help in fine-tuning the algorithm's parameters. After assuring the algorithm's convergence, a dynamic model of smoke will be implemented along with a fitness function that will allow a successful location of the source of the smoke. The basic PSO algorithm will then be applied in a confined space containing the smoke and its source. At this point, the algorithm may already need some modifications due to the specific nature of the measured fitness function.

Next, all constraints and limitations of a quadrotor UAV will be considered. This includes its non-zero dimensions, wind funnels created by rotors, and, mainly, the implicit inability to know its immediate position. For position tracking, several types of sensors will be implemented into the simulation. For relative localization of swarm members, a camera system that is currently being developed by the Department of Cybernetics will be used. For absolute vertical localization of the robots, barometric, ultrasonic or laser (or a combination of these) altimeters could prove effective, and for localization in the horizontal plane, a camera together with an algorithm computing the optical flow of the camera's picture will be used.

Eventually, the basic PSO algorithm will be modified to take into account the mentioned limitations. After functionality of this version of the algorithm reaches a sufficient level, it will be used to successfully locate the source of a smoke. The performance of the final algorithm will then be evaluated based on observation and compared to the performance of the basic version of the algorithm.

2 Related work

Paper [4] deals with the task of locating a plume source under water with an AUV (autonomous underwater vehicle). Two approaches at locating the source of a plume are presented. The first approach causes the vehicle to trace the plume to its source. The second approach uses an algorithm to estimate the location of the source based on measured concentration of the plume and water flow at different spots. It is stated that for environments with a low Reynolds number (ratio of inertial forces to viscous forces in a fluid) where laminar flow is prevalent, a gradient-type searching algorithm can be employed. However, for environments with a large amount of turbulences where the direction of fluid flow does not give correct information about the source of the plume, a gradient following approach is not suited. Since a plume with prevalent laminar flow will be considered in this work, the PSO algorithm seems to be a reasonable choice, because it is a type of gradient-following algorithm.

3 Simulation environment

Throughout this entire work, the operation area for the robot swarm will stay the same. It is a cuboid 16 meters wide, 16 meters deep and 9 meters high. The area contains no obstacles. In each plot showing the progress of an algorithm, one view from the top and two views from the side of the operation area are be shown. The robots' movement is not limited to this area, but the new positions calculated in each iteration of the PSO algorithm are be located within the boundaries of this cuboid. This is to assure all-time visibility of the swarm members in the plot, but allow occasional crossing of the boundaries due to the robots' inertia. An example of a plot of the operation area is in figure 1.

Figure 1: View of the simulation environment with 2 quadrotors

3.1 Fitness function

One of the main parts of the PSO algorithm is a fitness function, whose values serve as the input of the algorithm. Its value is a function of spatial coordinates (mainly) and time. The operation area is divided into a grid of cubical cells of small size. Each of these cells can contain a value calculated from the cell's x, y and z coordinates (and optionally time) using the fitness function's formula. The reason why a discrete cell grid instead of a continuous space is chosen to store the values is explained in chapter 6.2.

3.2 Used symbols

The following symbols are used in the visual representation of the simulation progress:

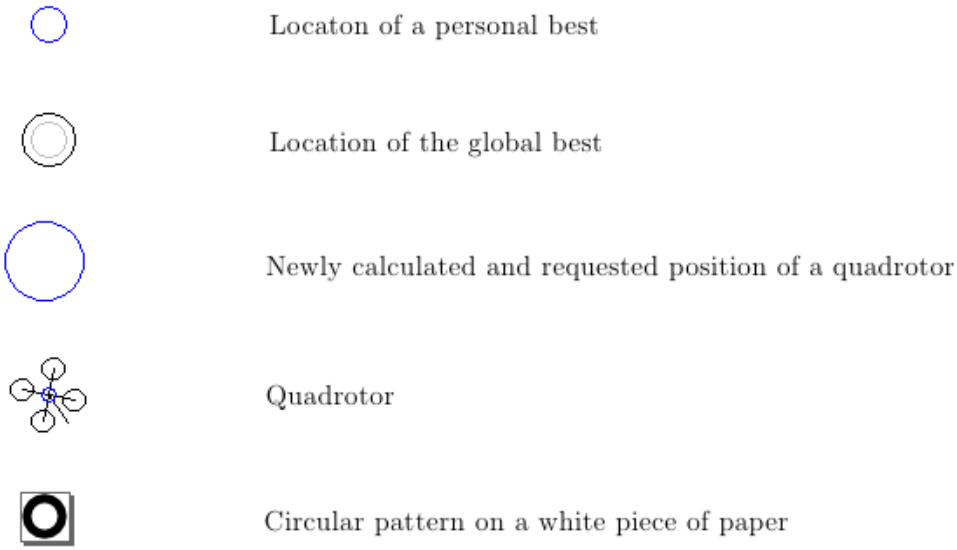

Figure 2: Used simulation symbols

4 Quadrotor helicopter model and regulator

Since smoke generally spreads both horizontally and vertically, the use of ground robots is not possible. A possible exception would be smoke released in an operation area of very low height, where the wind velocity and concentration of the gas at position $[x, y, 0]$ would be roughly equal to the values measured at position $[x, y, z_{max}]$. The robot type used to achieve the goal of this work has to be able to travel not only on the ground surface of the operation area, but also in the vertical direction.

The quadrotor helicopter is a type of aerial vehicle frequently used for robotic purposes. An example of a quadrotor can be seen in figure 3. The presence of four rotors, instead of a large one used in standard helicopters, allows the use of much smaller and cheaper propellers less prone to being damaged. In combination with a foam guarding frame surrounding the rotors, the quadrotor is a durable aerial vehicle well suited for robotic experiments where accidents are expected. Also, its symmetry allows for equipment installation on all four sides equally (for example cameras, special markers for recognition by other cameras, sensors etc.).

Figure 3: Quadrotor with a foam protection frame [1]

4.1 Dynamic model

To simulate realistic movement of the quadrotor, the dynamic model and regulator presented in [2] will be used. First, two reference frames are introduced to describe the state of the quadrotor, as seen in figure 4. The coordinate system $\{\vec{\mathbf{i}}_1, \vec{\mathbf{i}}_2, \vec{\mathbf{i}}_3\}$ is an inertial reference frame connected to the operation area of the robot. The coordinate system $\{\vec{\mathbf{b}}_1, \vec{\mathbf{b}}_2, \vec{\mathbf{b}}_3\}$ is connected to the body of the quadrotor with its origin in the center of mass of the quadrotor. The vectors $\vec{\mathbf{b}}_1$ and $\vec{\mathbf{b}}_2$ lie in the plane defined by the four centers of propellers, the $\vec{\mathbf{b}}_3$ vector is perpendicular to this plane. In later chapters, the $\vec{\mathbf{b}}_1$ vector is used to indicate the 'front' of the robot.

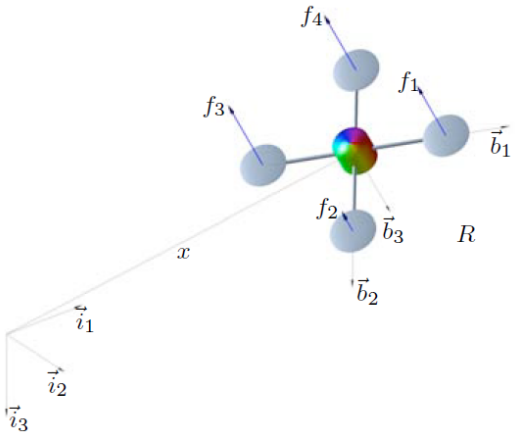

Figure 4: Quadrotor model with used reference frames [2]

Let us define the following variables [2]:

$m \in \mathbb{R}$	the total mass of the quadrotor
$J \in \mathbb{R}^{3\times3}$	the inertia matrix with respect to the body-fixed frame
$R \in \mathrm{SO}(3)$	the rotation matrix from the body-fixed frame to the inertial frame
$\vec{\Omega} \in \mathbb{R}^3$	the angular velocity in the body-fixed frame
$\vec{\mathbf{x}} \in \mathbb{R}^3$	the location of the quadrotor's center of mass in the inertial frame
$\vec{\mathbf{v}} \in \mathbb{R}^3$	the velocity of the quadrotor's center of mass in the inertial frame
$d \in \mathbb{R}$	the distance from the center of mass to the center of each rotor
$f_i \in \mathbb{R}$	the thrust generated by the i-th propeller along the $-\vec{b_3}$ axis
$\tau_i \in \mathbb{R}$	the torque generated by the i-th propeller about the $-\vec{b_3}$ axis
$f \in \mathbb{R}$	the total thrust ($\Sigma_{i=1}^4 f_i$)
$M \in \mathbb{R}$	the total moment in the body-fixed frame

4.2 Regulator

The following equations describe the quadrotor's motion:

$$\begin{aligned}
\dot{\vec{\mathbf{x}}} &= \vec{\mathbf{v}} && (1)\\
m\dot{\vec{\mathbf{v}}} &= mg\vec{\mathbf{e}}_3 - fR\vec{\mathbf{e}}_3 && (2)\\
\dot{R} &= R\hat{\Omega} && (3)\\
J\dot{\Omega} + \Omega \times J\Omega &= M && (4)
\end{aligned}$$

First, the position and velocity tracking errors are calculated:

$$\vec{\mathbf{e}}_x = \vec{\mathbf{x}} - \vec{\mathbf{x}}_d \quad (5)$$
$$\vec{\mathbf{e}}_v = \vec{\mathbf{v}} - \vec{\mathbf{v}}_d \quad (6)$$
$$(7)$$

where the subscript d denotes the *desired* value of a variable. The desired direction of the $\vec{b}_3$ vector is calculated:

$$\vec{\mathbf{b}}_{3d} = -\frac{-k_x\vec{\mathbf{e}}_x - k_v\vec{\mathbf{e}}_v - mg\vec{\mathbf{e}}_3}{|-k_x\vec{\mathbf{e}}_x - k_v\vec{\mathbf{e}}_v - mg\vec{\mathbf{e}}_3|} \quad (8)$$

Using the $\vec{\mathbf{b}}_{1d}$ vector, which is also the regulator's input, $\vec{\mathbf{b}}_{2d}$ and R_d are calculated:

$$\begin{aligned}
\vec{\mathbf{b}}_{2d} &= \frac{(\vec{\mathbf{b}}_{3d} \times \vec{\mathbf{b}}_{1d})}{\left|\vec{\mathbf{b}}_{3d} \times \vec{\mathbf{b}}_{1d}\right|} && (9)\\
R_d &= \left[\vec{\mathbf{b}}_{2d} \times \vec{\mathbf{b}}_{3d}, \vec{\mathbf{b}}_{2d}, \vec{\mathbf{b}}_{3d}\right] && (10)
\end{aligned}$$

Then, the rotation matrix and angular velocity matrix errors are calculated:

$$\begin{aligned}
\vec{\mathbf{e}}_R &= \frac{1}{2}(R_d^T R - R^T R_d)^\vee && (11)\\
\vec{\mathbf{e}}_\Omega &= \vec{\Omega} - R^T R_d \vec{\Omega}_d && (12)
\end{aligned}$$

Finally, the total force and moments are calculated:

$$
\begin{aligned}
f &= -(-k_x\vec{\mathbf{e}}_x - k_v\vec{\mathbf{e}}_v - mg\vec{\mathbf{e}}_3)R\vec{\mathbf{e}}_3 && (13)\\
M &= -k_R\vec{\mathbf{e}}_R - k_\Omega\vec{\mathbf{e}}_\Omega + \vec{\Omega}\times J\vec{\Omega} - J(\hat{\vec{\Omega}}R^TR_d\vec{\Omega}_d - R^TR_d\dot{\vec{\Omega}}_d) && (14)
\end{aligned}
$$

To limit the maximum thrust of the propellers, the following equation is used:

$$
\begin{pmatrix} f \\ M_1 \\ M_2 \\ M_3 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & -d & 0 & d \\ d & 0 & -d & 0 \\ -c_{\tau f} & c_{\tau f} & -c_{\tau f} & c_{\tau f} \end{pmatrix} \cdot \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{pmatrix} \tag{15}
$$

Values f_1 through f_4 are expressed from the equation, attenuated according to the maximum thrust of each propeller, and plugged back into the equation to obtain the final force and moments. The new velocity, position and rotation of the quadrotor are then calculated. Since the regulator calculates the trajectory's discrete steps, step time Δt is defined:

$$
\begin{aligned}
\vec{\mathbf{v}}_{new} &= \frac{\Delta t}{m}(mg\vec{\mathbf{e}}_3 - fR\vec{\mathbf{e}}_3) + \vec{\mathbf{v}} && (16)\\
\vec{\mathbf{x}}_{new} &= \vec{\mathbf{v}}_{new} + \vec{\mathbf{x}} && (17)\\
\vec{\Omega}_{new} &= J^{-1}(M - \vec{\Omega}\times J\vec{\Omega}) && (18)\\
R_{new} &= RR_\omega && (19)
\end{aligned}
$$

where

$$
R_\omega =
$$

$$
\begin{pmatrix}
1 + (1-\cos(a))(\omega_x^2 - 1) & \omega_z\sin(a) + (1-\cos(a))\omega_x\omega_y & \omega_y\sin(a) + (1-\cos(a))\omega_x\omega_z \\
-\omega_z\sin(a) + (1-\cos(a))\omega_x\omega_y & 1 + (1-\cos(a))(\omega_y^2 - 1) & -\omega_x\sin(a) + (1-\cos(a))\omega_y\omega_z \\
-\omega_y\sin(a) + (1-\cos(a))\omega_x\omega_z & \omega_x\sin(a) + (1-\cos(a))\omega_y\omega_z & 1 + (1-\cos(a))(\omega_z^2 - 1)
\end{pmatrix}
$$

$$
\begin{aligned}
\omega_x &= \Omega(1)\\
\omega_y &= \Omega(2)\\
\omega_z &= \Omega(3)\\
a &= \Delta t \cdot |\Omega|
\end{aligned}
$$

The used constants have the following values:

parameter	value	units
m	4.34	kg
k_x	6.32	–
k_v	7.6	–
k_R	19.81	–
k_Ω	2.54	–
g	-9.81	$m \cdot s^{-2}$
Δt	0.005	s

$$J = \begin{pmatrix} 0.082 & 0 & 0 \\ 0 & 0.0845 & 0 \\ 0 & 0 & 0.1377 \end{pmatrix}$$

5 Means of localization

In order for the swarm to navigate correctly through the operation environment, a means of absolute and relative localization of the swarm members must be present. This is for three reasons:

- In order to calculate the trajectories of the swarm members, the PSO algorithm needs to know their absolute position (position in the operation environment's reference frame) and the position of their measured personal best value of the fitness function.
- The members need to be able to tell the location of the located target in the operation area's reference frame.
- The robots must have the ability to detect imminent collisions with the environment and themselves, and avoid such collisions.

5.1 Absolute localization

Not only need the robots know their position in the environment, they also need to have information about their complete state. That includes the rotation of the body-fixed reference frame in the inertial reference frame. Several types of sensors may serve as sources of information about the state of the quadrotor:

- An **optical flow sensor** pointed towards the ground could serve as a sensor of displacement in the horizontal plane of the inertial reference frame. To compensate for the quadrotor's pitch and roll, the sensor could be mounted on a (motorized) ball pivot.
- In addition to the optical flow sensor, **accelerometers** could be mounted to measure the robot's movement.
- To calculate the direction of the $\vec{\mathbf{b}}_1$ vector, a **gyroscope** (or multiple gyroscopes) could be mounted on the quadrotor. An alternative to this would be an accelerometer attached in a place far from the robot's center of mass. That way, the acceleration of a point on the circumference of the robot would be measured, which would, after double integration, yield the robot's rotation about the axis given by $\vec{\mathbf{b}}_3$.
- To measure the altitude of the robot, a **barometric altimeter** (inaccurate) in combination with an **ultrasonic** or **laser altimeter** could be used.

The task of acquiring reliable means of the quadrotor's absolute localization is a complex one. The sensors mentioned earlier for horizontal position measuring produce additive errors, which would after a period of time lead to a difference between the actual robot's

position and the position calculated by the robot itself. The measured altitude of the robot would be altered by any object on the ground of the operation area, which would lead to rapid altitude changes of the robot (in effort to maintain a desired altitude above a measured point). All the errors above mentioned could be reduced using relative camera localization of the robots and fusion of data from multiple cameras.

For the sake of the goal this work aims to achieve, let us assume that the robots know their absolute position and rotation with precision. Also, the operation area will not contain any obstacles or walls, and the ground will be flat.

5.2 Relative localization

At the CTU's Department of Cybernetics, a camera system for pattern recognition and localization is currently in development. An emulation of this system is implemented in the simulation environment for this work. The system is depicted in figure 5.

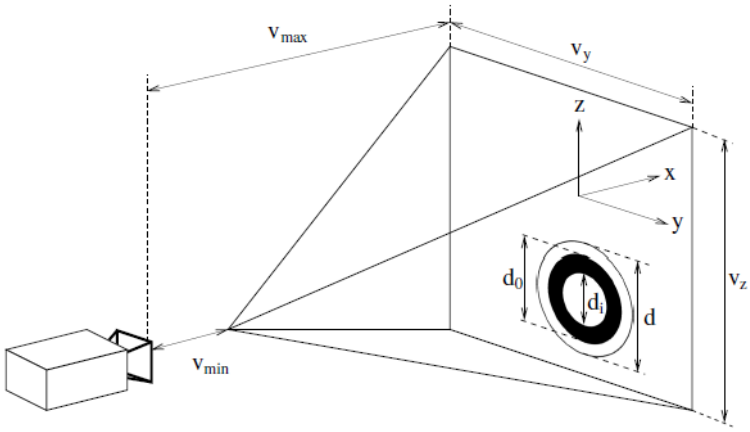

Figure 5: The camera setup with its field of view and a pattern [3]

The system uses an algorithm that scans images received from the camera for black circular patterns on a white background. These patterns are called *blobs*. It is then able to determine the distance of the pattern from the camera from the size of the its principal axis in the captured picture. The relative position of the blob in the yz plane (see figure 5) is calculated from the position of the blob in the captured picture.

The field of view of the camera can be represented by a pyramid with its apex closer to the camera. The camera is placed with an offset of 20 cm from the center of the quadrotor along the $\vec{\mathbf{b}}_1$ vector and $v_{min} = 5$ cm. The properties of vector dot product are used to determine whether a blob is visible by the camera. The following conditions must be true in order for the blob to be considered visible (for symbol definitions, see figures 5 and 6):

$$\frac{\vec{\mathbf{d}} \cdot \vec{\mathbf{b}}_1}{|\vec{\mathbf{d}}||\vec{\mathbf{b}}_1|} > 0 \tag{20}$$

$$\left||\vec{\mathbf{d}}|\cos(\alpha)\right| <= v_{max} - v_{min} \tag{21}$$

$$\left||\vec{\mathbf{d}}|\cos(\beta)\right| <= \left||\vec{\mathbf{d}}|\cos(\alpha)\right| \frac{w_{max}}{v_{max} - v_{min}} \tag{22}$$

$$\left||\vec{\mathbf{d}}|\cos(\gamma)\right| <= \left||\vec{\mathbf{d}}|\cos(\alpha)\right| \frac{h_{max}}{v_{max} - v_{min}} \tag{23}$$

$$\frac{\vec{\mathbf{n}}_{blob} \cdot \vec{\mathbf{b}}_1}{|\vec{\mathbf{n}}_{blob}||\vec{\mathbf{b}}_1|} < \cos(\pi - blobAngle_{max}) \tag{24}$$

where $blobAngle_{max}$ is the maximum angle in degrees allowed betwen the $\vec{\mathbf{n}_{blob}}$ vector and the yz plane's normal vector (pointing towards the camera).

Equation 20 guarantees that the blob is in front of the camera and 'below' the pyramid's apex. Equation 21 limits the blob's distance from the camera. Equations 22 and 23 guarantee the blob is within the pyramid's boundaries in the yz plane. And finally, equation 24 only makes the blob visible if the angle between its normal vector and the yz plane's normal vector is less than $blobAngle_{max}$.

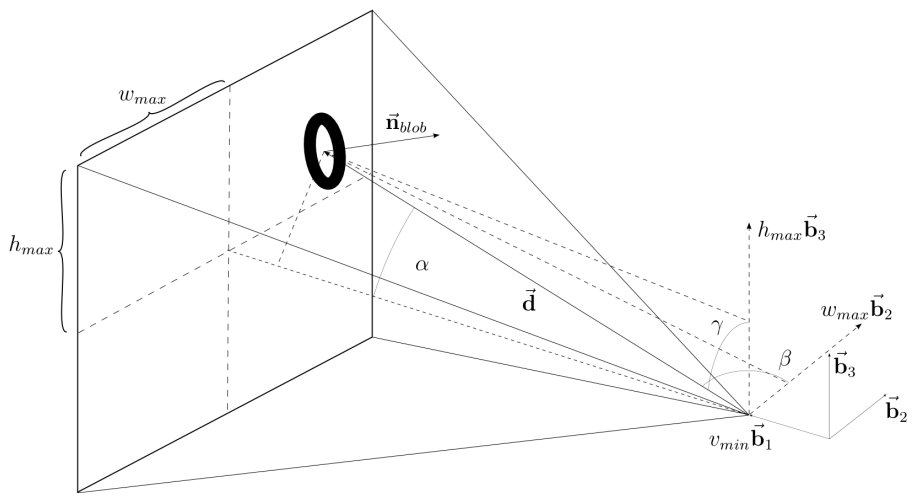

Figure 6: The camera and blob properties used to determine the blob's visibility

If all the above conditions are satisfied, the blob is considered visible for the quadrotor owning the camera, and its exact position is know by this quadrotor. The system allows for either the detection of a single blob or multiple blobs. Also, if the blobs are equipped with an additional unique marking symbol, the camera system is able to distinguish them. Some of these possibilities are exploited in chapter 7.

6 Odor model

For the specific application of the PSO algorithm this work deals with, the localization of an odor source, a sufficiently accurate dynamic model of smoke has to be implemented. That includes the movement of emitted particles, the effect of wind acting upon the particles and possibly changes from laminar to turbulent air flow. Since the algorithm aims to locate a point or a small area in the operation space where the value of the cost function has its maximum (or minimum), a smoke source of small dimensions has to be used. Therefore, after the algorithm is finished, it will not be applicable on large scale smoke sources such as wildfires, but on locating smaller smoke sources (gas leakage through a hole in a pipe). In this version of the simulation, the smoke source has zero dimensions, which means the particles are all emitted from the same point in the operation space.

6.1 Smoke particle

An individual smoke particle has the following parameters:

- [x, y, z] - position coordinates in the operation space
- $[v_x, v_y, v_z]$ - velocities along the respective axes of the operation space coordinate system
- $[v_{x_{init}}, v_{y_{init}}, v_{z_{init}}]$ - initial velocities at the moment of emission
- lifetime - the time passed since the particle's emission (more precisely the number of position changes the particle has undergone)

All the particles share the same value of a parameter called maximum lifetime, which says how many position changes each particle can go through before it is recycled. This is explained in chapter 6.3.

6.2 Smoke density grid

Since gas concentration sensors are used by the robots to 'measure' the smoke in different parts of the area, a means of determining the smoke density at a given point in space has to be devised. Because the smoke consists of a massive amount of particles (thousands to tens of thousands), the method must not be computationally complex. One way would be to calculate the distance of each particle from the sensor and compare it to a threshold value. The number of particles with distance smaller than this threshold value would then be considered as the smoke density at that point. This would, however, cause a significant load to the computer and decrease the framerate of the simulation.

As mentioned in chapter 3.1, a grid of discrete cubical cells filling the operation area is used to store the values of the cost function. The density at the location of the smoke sensor will simply be represented by the number of particles enclosed in the same cell where the sensor is located. As each particle leaves a cell, the cell's value will decrease by one. As it enters a new cell, the new cell's value will increase by one. A visual example is in figure 7.

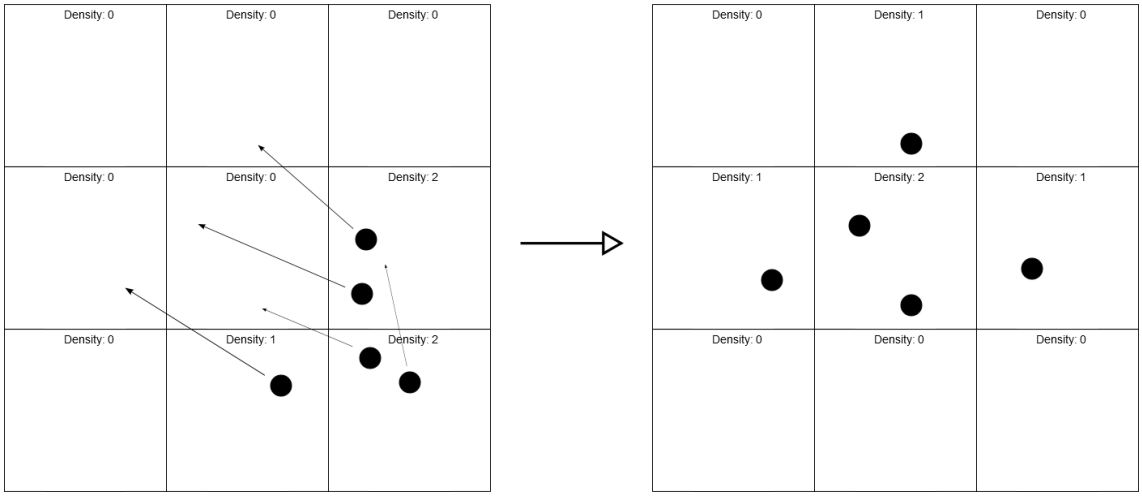

Figure 7: 2D analogy of the implemented smoke concentration mechanism

6.3 Smoke particle dynamics

The last step is to create a set of rules for the particles to obey in order to sufficiently resemble real smoke. Smoke in an air-filled area consists of small solid particles being carried away by molecules of air, or simply by molecules of a gas other than air.

Once emitted, these molecules are accelerated by moving molecules of air around them (wind). If a wind of constant velocity is considered, the smoke particles are acted upon by a constant force, which causes constant velocity increments in each discrete time step. In each time step, the particle is accelerated until it matches the velocity of the wind at its location. Additionally, a small random vector is added to the particle's calculated velocity vector to break the smoke's uniformity.

The simulation can only deal with a limited number of particles. To avoid the need of constantly creating new particles and dismissing old ones, particles are recycled. In the initial stage, new particles are emitted into the area. Once the maximum allowed number of particles is reached, no new particles are released, and the oldest ones are relocated back at the location of the smoke source with their initial velocity. If particles are released from the source in waves, the maximum number of steps a particle can undergo is given by $maxParticles/particleWave$, where $particleWave$ is the number of particles emitted in each time step. This effectively yields a continuous smoke. Algorithm 1 shows the policy of

smoke particle movement. A visualisation of the smoke in the operation area can be seen in figure 8.

```
For a smoke source located at [s_x, s_y, s_z] :
if existingParticles < maxParticles then
    for i = 0; i ≤ particleWave do
        create new particle;
        densityGrid[x_s, y_s, z_s] + +;
        existingParticles + +;
foreach existing particle p do
    if undergoneSteps >= maxAllowedSteps then
        densityGrid[p_x, p_y, p_z] − −;
        p_x ← s_x;
        p_y ← s_y;
        p_z ← s_z;
        densityGrid[s_x, s_y, s_z] + +;
    else
        if v_x ≠ wind_x then
            increase or decrease v_x by v_increment;
        if v_y ≠ wind_y then
            increase or decrease v_y by v_increment;
        if v_z ≠ wind_z then
            increase or decrease v_z by v_increment;
        v_x ← v_x + rand(0,1) · randomIncrement − rand(0,1) · randomIncrement;
        v_y ← v_y + rand(0,1) · randomIncrement − rand(0,1) · randomIncrement;
        v_z ← v_z + rand(0,1) · randomIncrement − rand(0,1) · randomIncrement;
        densityGrid[p_x, p_y, p_z] − −;
        p_x ← p_x + v_x;
        p_y ← p_y + v_y;
        p_z ← p_z + v_z;
        densityGrid[p_x, p_y, p_z] + +;
        undergoneSteps + +;
```

Algorithm 1: Algorithm of a single smoke step

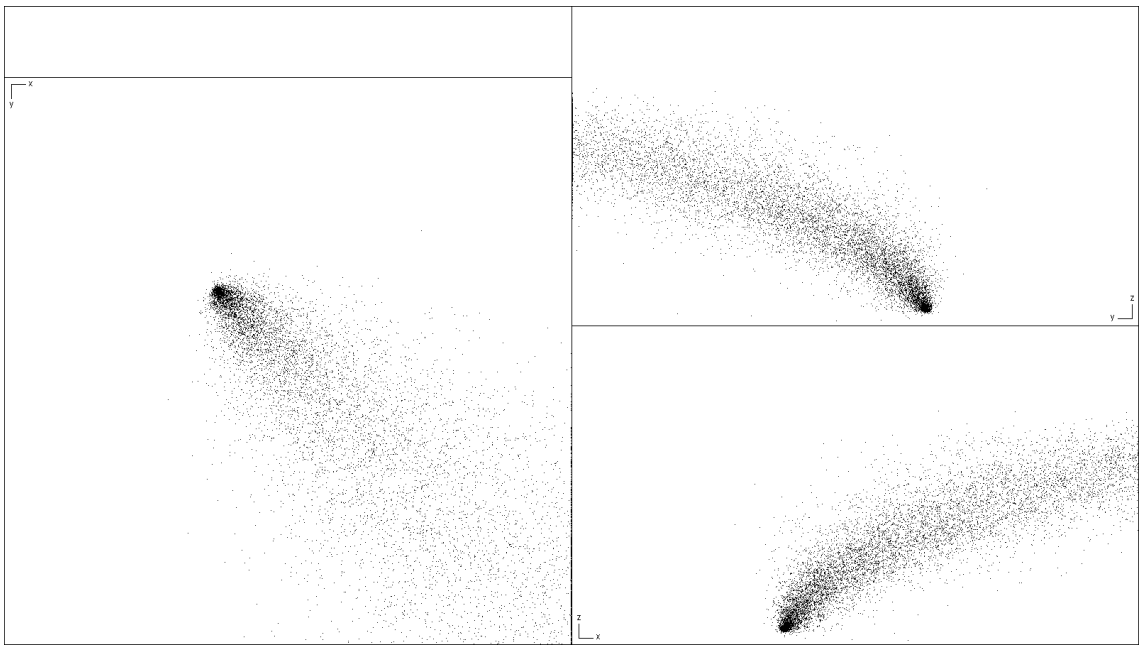

Figure 8: Simulation environment with one smoke source

7 PSO algorithm implementations

In this chapter, the PSO algorithm is modified using the localization means described in chapter 5 and the dimensionless swarm particles are replaced by the quadrotor. Instead of linear movement, the swarm members follow the rules of the quadrotor's dynamic model. In this work, the operation space only contains one global maximum of the fitness function. No obstacles except the boundaries of the operation space are considered.

The PSO algorithm aims at locating a global maximum of a function in an area using a group of particles called a swarm. In each iteration, a new position is calculated for each particle based on the values measured by the particle itself and all the other swarm members. The new position of a particle is calculated using the following formulas:

$$\vec{\mathbf{v}}_{i+1} = w \cdot \vec{\mathbf{v}}_i + c_1 \cdot r_1 \cdot (\vec{\mathbf{p}} - \vec{\mathbf{x}}_i) + c_2 \cdot r_2 \cdot (\vec{\mathbf{g}} - \vec{\mathbf{x}}_i) \tag{25}$$

$$\vec{\mathbf{x}}_{i+1} = \vec{\mathbf{x}}_i + \vec{\mathbf{v}}_{i+1} \tag{26}$$

where:

$\vec{\mathbf{v}}_{i+1} \in \mathbb{R}^3$	the position change to occur in iteration i+1
$\vec{\mathbf{v}}_i \in \mathbb{R}^3$	the position change that occured in iteration i
$\vec{\mathbf{x}}_{i+1} \in \mathbb{R}^3$	the particle's position in iteration i+1
$\vec{\mathbf{x}}_i \in \mathbb{R}^3$	the particle's position in iteration i
$\vec{\mathbf{p}} \in \mathbb{R}^3$	the position of the best fitness function value measured by this particle (personal best)
$\vec{\mathbf{g}} \in \mathbb{R}^3$	the position of the best fitness function value measured by all particles (global best)
$w \in \mathbb{R}$	the 'inertia weight' coefficient
$c_1 \in \mathbb{R}$	the 'personal best' coefficient
$c_2 \in \mathbb{R}$	the 'global best' coefficient
$r_1 \in \mathbb{R}$	a random number in the interval $< 0; 1 >$
$r_2 \in \mathbb{R}$	a random number in the interval $< 0; 1 >$

The presence of the $(\vec{\mathbf{g}} - \vec{\mathbf{x}})$ vector prevents the particles from oscillating around the position of a local minimum (the particle's personal best) and eventually stopping there. This doesn't apply if the position of the particle's personal best is equal to the position of the global best, but that is always the case of only one swarm member.

7.1 Basic PSO algorithm

7.1.1 Simulation setup

The first in a series of steps to achieve the final goal is to create a basic PSO simulation and tune its parameters for optimal performace. In this version, the simulation setup is following:

- Robots considered as dimensionless and incapable of collisions
- Robots know their absolute position
- Movement using the quadrotor dynamic model and regulator
- Simple time-invariant fitness function
- Basic version of PSO algorithm
- Continuous fitness function sensor reading every 0.3 seconds (not only at positions calculated by the algorithm)

Since it is presumed that only one global maximum exists, the parameters of the algorithm are tuned accordingly. The values the experiments and tuning started at are $c_1 = 2$ and $c_2 = 2$ as recommended in [5]. After several experiments, the selected values are:

parameter	value	units
w	0.9	–
c_1	1.2	–
c_2	2	–

Table 1: Parameter values of the basic PSO algorithm

This forces the robots to prefer the direction of travel given by the location of the global best. Also, the $\vec{\mathbf{v}}_0$ vector is set non-zero to avoid stagnation at start.

7.1.2 Fitness function

The fitness function designed for this simulation is computed in each cell of the density grid using the following formula:

$$\frac{10^3}{(1+\sqrt{(\frac{x-g_x}{5000})^2+(\frac{y-g_y}{5000})^2+(\frac{z-g_z}{5000})}}$$

where $[g_x, g_y, g_z]$ is the desired location of the global maximum. If the locations of all personal bests and the global best are a certain threshold distance away from the true location of the global maximum (1 m), the run is considered successful. In the visual representation of the simulation, each view of the operation area has a background colored according to the values of the fitness function in a cross section of the operation area parallel to the view direction. On the axis perpendicular to the view plane, the cross section has the same coordinate as the global maximum.

7.1.3 Simulation results

The location of the global maximum is set to $[12, 11, 2]\,[m]$. The starting area of the quadrotors is in the opposite corner of the operation area. The progress of a single run of the algorithm can be seen in figure 9. The quadrotors have successfully located the global maximum after 7 iterations.

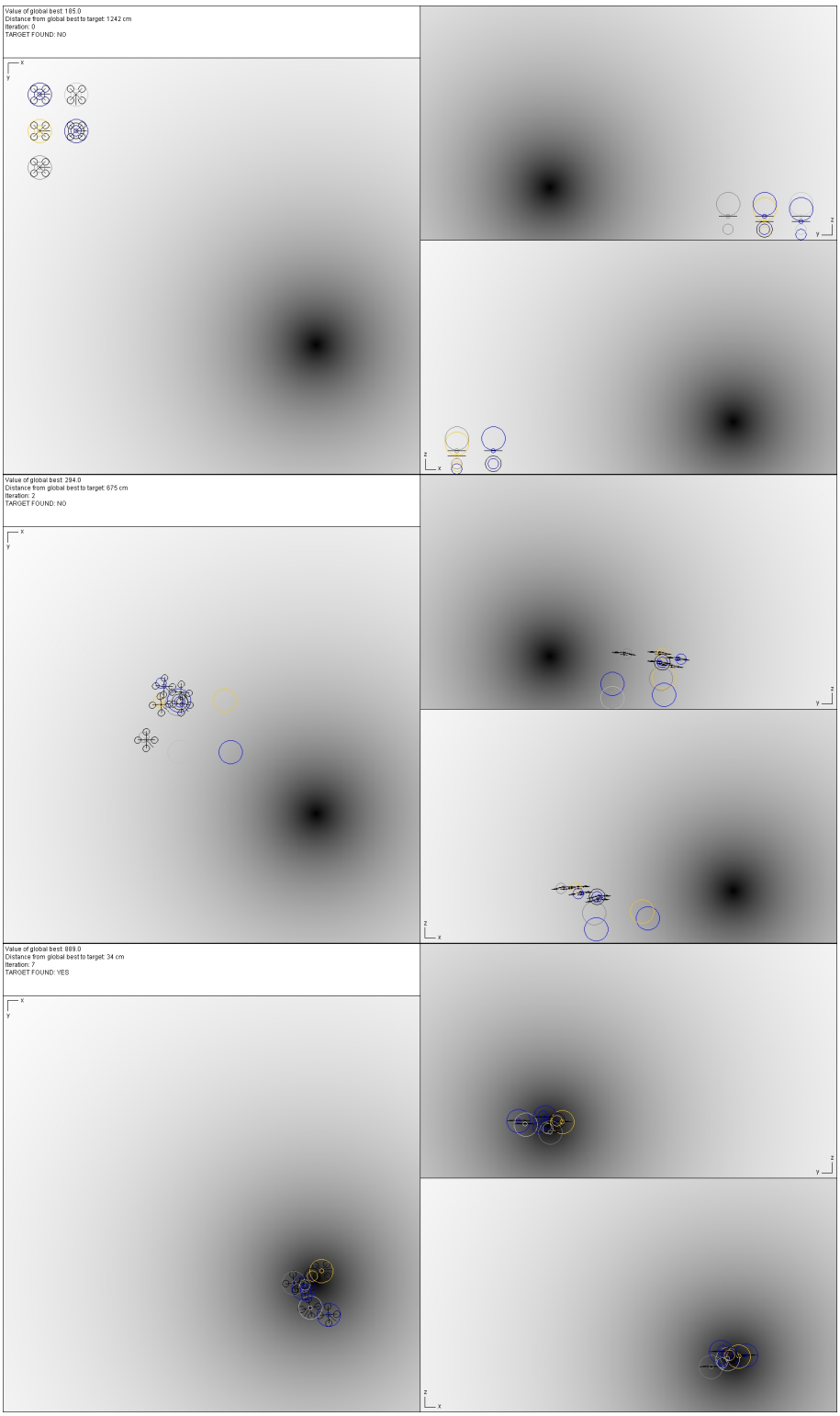

Figure 9: Basic PSO algorithm with a simple fitness function

7.2 Basic PSO algorithm for smoke source localization

7.2.1 Simulation setup

In this simulation, a smoke source is added into the operation area. The simulation has these parameters:

- Robots considered as dimensionless and incapable of collisions
- Movement using the quadrotor dynamic model and regulator
- Fitness function based on smoke particles concentration
- PSO algorithm modified for the specific application of locating a smoke source
- Continuous fitness function sensor reading every 0.3 seconds (not only at positions calculated by the algorithm)

7.2.2 Fitness function

The fitness function from the previous chapter is replaced by the concentration of particles at a specific point in the area. Precisely, it is the number of particles enclosed in a single density grid cell. Since the maximum number of particles is limited to 8000, measurements far from the actual source almost always show zero, even though there are particles in close vicinity of the sensor. This is fixed by enlarging the scope of the sensor. The sensor then measures the concentration in all adjacent cells within the area of its scope and adds it up. The scope is given by:

$$< x - range; x + range > \times < y - range; y + range > \times < z - range; z + range >$$

The used value for $range$ is 5 [grid cell].

7.2.3 PSO algorithm modifications

Because the nature of the fitness function is known beforehand, the algorithm can be modified to help find the global maximum more efficiently. Two modifications are introduced:

Blind searching Since areas not containing any smoke particles have a zero fitness value function (the starting area of the swarm, for instance), the algorithm couldn't evolve without an initial blind search. For each swarm member, a random new position within 3 meters of the robot's current position is calculated in each iteration until a non-zero concentration is found. The algorithm then switches to PSO mode and starts locating the smoke source.

Forced personal best relocation Once the the algorithm switches to PSO mode, there still may be swarm members with a zero value of personal best. To avoid their returns to the positions of their personal best (which is of no value to the algorithm's progress), the location of their personal best is forcibly changed to a position within 4 meters of the global best (the value is left zero). This allows all the members to explore a more prospective area of the operation environment.

7.2.4 Smoke source location prediction

Numerous expertiments using the 2 modifications mentioned above were performed, but successful goal location was not always achieved due to the specific nature of the fitness function. To help compensate for the fact that not the complete operation area is filled with non-zero fitness function values, the position of the global maximum can be predicted and used for faster convergence of the algorithm.

The algorithm stores the last n values of global best (instead of one). It then calculates a vector pointing from the location of the best global best towards the estimated smoke source location:

$$\vec{\mathbf{s}}_{estimate} = \vec{\mathbf{g}_1} + k1\frac{\vec{\mathbf{g}_1} - \vec{\mathbf{g}_n}}{|\vec{\mathbf{g}_1} - \vec{\mathbf{g}_n}|} \tag{27}$$

where subscript 1 denotes the newest global best with the highest value.

If a preceding knowledge of the nature of the gas or smoke is assumed, the source location prediction can be made more accurate. For instance, if the leaking gas has a density lower than air, it will travel vertically upwards. If the robots aren't right above the ground and detect the gas, there is a high chance the smoke source is located in a lower region. Using this assumption, equation 27 can be modified:

$$s_x = g_x + k1\frac{g_{1x} - g_{nx}}{|\vec{\mathbf{g}_1} - \vec{\mathbf{g}_n}|} \tag{28}$$

$$s_y = g_y + k1\frac{g_{1y} - g_{ny}}{|\vec{\mathbf{g}_1} - \vec{\mathbf{g}_n}|} \tag{29}$$

$$s_z = g_z - k2 \tag{30}$$

This will always place the predicted location of the smoke source lower than the global best, forcing the swarm to search lower parts of the operation area. The last modification is an introduction of an upper boundary for new position calculations by the PSO algorithm. Once a global best has been found, the maximum z-coordinate any newly calculated quadrotor position can have is the global best's z-position.

The PSO algorithm is modified to contain the predicted location of the smoke source:

$$\vec{\mathbf{v}}_{i+1} = w \cdot \vec{\mathbf{v}}_i + c_1 \cdot r_1 \cdot (\vec{\mathbf{p}} - \vec{\mathbf{x}}_i) + c_2 \cdot r_2 \cdot (\vec{\mathbf{g}} - \vec{\mathbf{x}}_i) + c_3 \cdot r_3 \cdot (\vec{\mathbf{s}}_{estimate} - \vec{\mathbf{x}}_i) \tag{31}$$

Variables c_3 and r_3 have the same properties as c_1, c_2, r_1 and r_2.

The values of all the parameters for this simulation were experimentally set as follows:

parameter	value	units
w	0.4	–
c_1	0.4	–
c_2	0.55	–
c_3	1	–
k_1	4	m
k_2	3.2	m

Table 2: Parameter values of the basic smoke source searching PSO algorithm

7.2.5 Simulation results

The smoke source is placed at $[14, 12, 0.4]\,[m]$. The direction of the wind is set such that the smoke travels towards the corner at $[0, 0, 9]\,[m]$. The robots' starting position is the same as in the previous chapter. The progress of the modified algorithm can be seen in figure 10. The quadrotors have located the smoke source after about 18 iterations, with a noticeable stagnation in the middle of the progress.

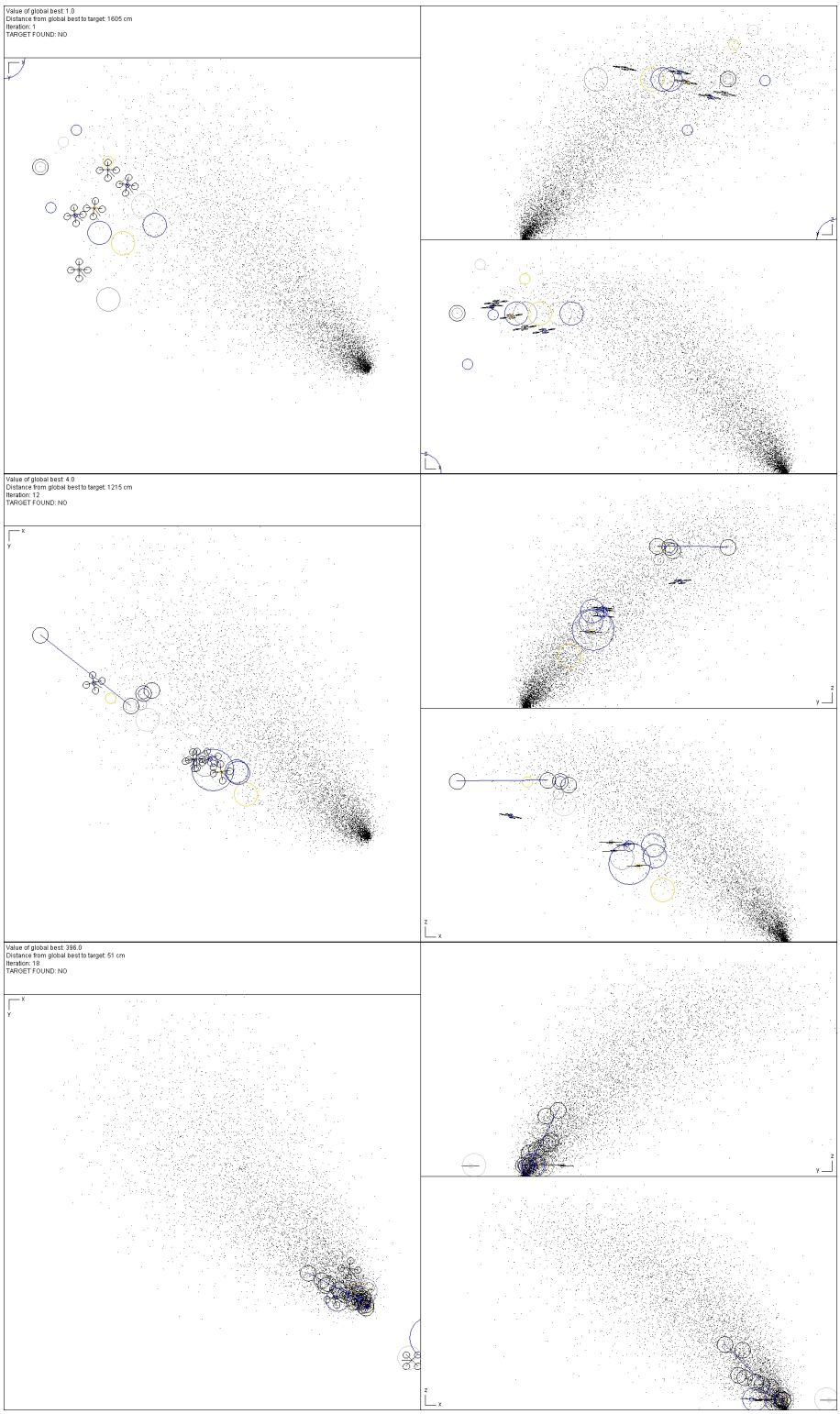

Figure 10: PSO algorithm modified for locating a smoke source

7.3 PSO algorithm with collision avoidance

7.3.1 Simulation setup

In this simulation, the dimensions and other factors linked to real robots are considered and implemented into the algorithm:

- Robot dimensions taken into account
- Movement using the quadrotor dynamic model and regulator
- Simple time-invariant fitness function
- PSO algorithm modified to avoid collisions of swarm members
- Continuous fitness function sensor reading every 0.3 seconds (not only at positions calculated by the algorithm)

7.3.2 Fitness function

The fitness function introduced in chapter 7.1 is used.

7.3.3 PSO algorithm modifications

As mentioned in chapter 5.1, it is assumed that the robots know their absolute position with respect to the inertial reference frame of the operation area. To locate other swarm members and avoid collisions, the robots use the camera system described in chapter 5.2. Also, the position error calculated by the regulator is limited by a threshold value to avoid rapid movement of the quadrotors. Three new modifications are implemented:

Non-colliding position calculation This modification covers two problems. The first problem is that the wind funnels created by the quadrotor's propellers would destabilize any other quadrotor flying below it. Therefore, the new positions calculated by the PSO algorithm must not be located above each other. Secondly, the new positions must be a certain distance from each other, otherwise the quadrotors could not occupy them without colliding.

The first problem reduces the task of calculating new positions to two dimensions, because the space occupied by a quadrotor is approximated by a vertically placed cylinder with height same as the height of the operation area. Algorithm 2 describes the way non-colliding new positions are achieved in each iteration.

Relative height differece limitation To allow the robots to see other members of the swarm when facing them, a maximum allowed difference of their z-coordinates must be introduced due to the limitations of the camera's field of view. Therefore, each new calculated position of a robot must be within a certain range along the z-axis from all the other robots.

Collision avoidance In addition to the two previous modifications, the policy for the movement of swarm members is changed. At any time, only one robot is allowed to travel, while the other remain stationary. An algorithm constantly checks whether the robots have either traveled to their requested location or are blocked by other robots. If all robots are blocked or at their requested position, a new iteration of the PSO algorithm is started. This may cause the position change calculated by the PSO algorithm and the actual position change of a robot to differ. Before calculating a new iteration, the $\vec{\mathbf{v}}_i$ value of each robot is replaced be the actual value.

```
P ← ∅;    // set of points occupied by newly calculated robot positions
// Only the x and y coordinates of vectors are considered, z = 0.
foreach quadrotor q do
    calculate new position r⃗ using basic PSO;
    cycleCount ← 0;
    collisionCount ← 1;
    while collisionCount > 0 do
        collisionCount ← 0;
        foreach occupied position p ∈ P do
            if |r⃗_pos − p⃗_pos| = 0 then
                r⃗_pos ← r⃗_pos + minDist (q⃗_pos − r⃗_pos)/|q⃗_pos − r⃗_pos|;
                collisionCount++;
            else
                if |r⃗_pos − p⃗_pos| < minDist then
                    r⃗_pos ← p⃗_pos + minDist (r⃗_pos − p⃗_pos)/|r⃗_pos − p⃗_pos|;
                    collisionCount++;
            if collisionCount > 0 then
                if r_x <lower x boundary then
                    r_x ← r_x + 2(p_x − r_x);
                else
                    if r_x >upper x boundary then
                        r_x ← r_x − 2(p_x − r_x);
                if r_y <lower y boundary then
                    r_y ← r_y + 2(p_y − r_y);
                else
                    if r_y >upper y boundary then
                        r_y ← r_y − 2(p_y − r_y);
        cycleCount++;
                                                    // Infinite loop avoidance
        if cycleCount > 30 then
            push r⃗_pos in the direction away from the occupied point last used in
            calculations for a distance of 3 · minDist;
            cycleCount ← 0;
    P ← P ∪ r⃗_pos
```

Algorithm 2: Algorithm for calculating non-colliding new positions of robots

The constants of the algorithm were tuned as follows:

parameter	value	units
w	0.2	–
c_1	1.4	–
c_2	1.6	–
$minDist$	140	cm
v_{max}	5	m
w_{max}	5	m
h_{max}	5	m

Table 3: Parameter values of the PSO algorithm with collision avoidance

7.3.4 Simulation results

This version of the algorithm converges significantly slower (in terms of time, not number of iterations) due to the inability of the robots to travel simultaneously. By tuning the algorithm's parameters, it can be sped up, but the quadrotors are then more prone to colliding. A progression of this algorithm is seen in figure 11. The top view of the area in this figure includes the pyramidal field of view of the robots' cameras.

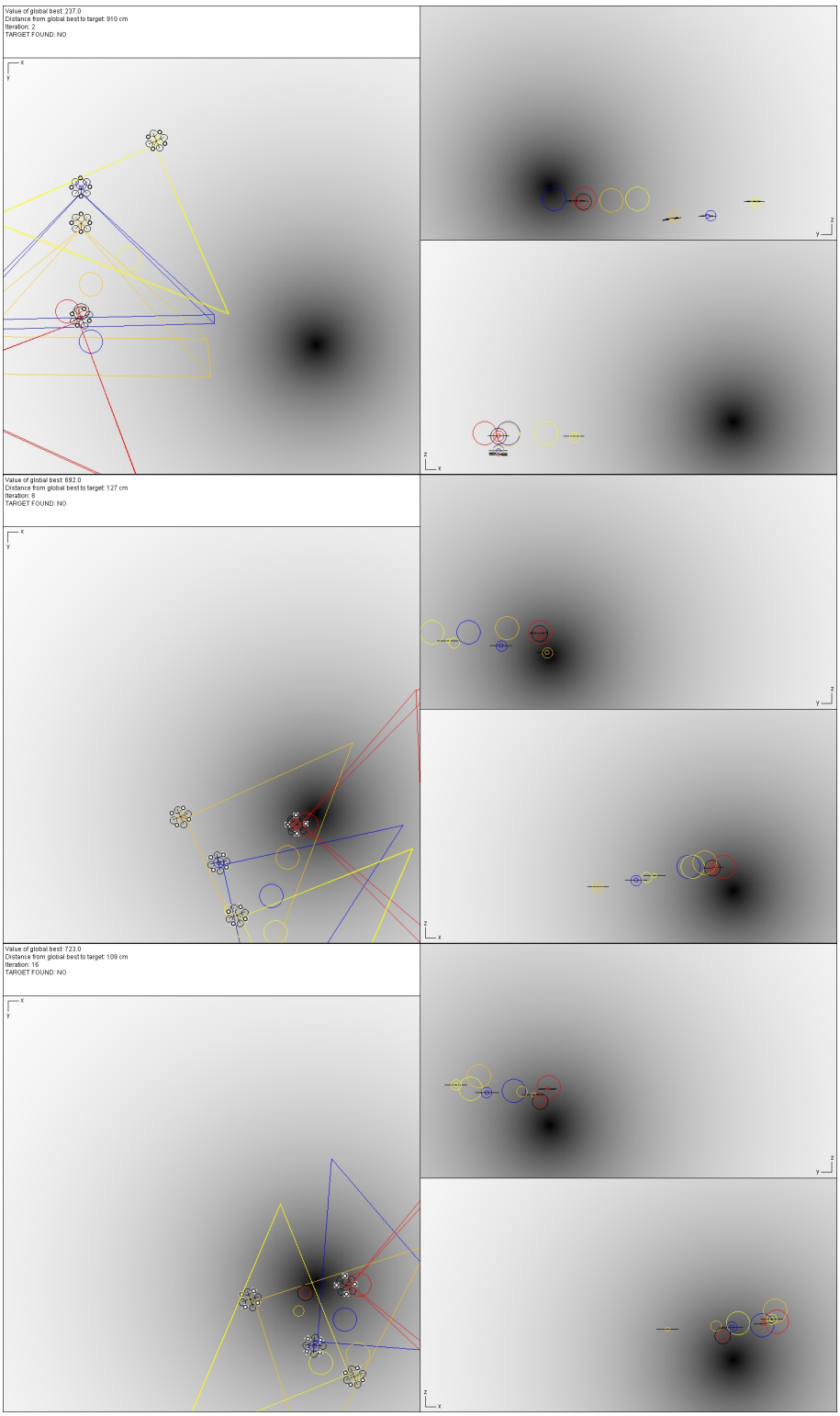

Figure 11: PSO algorithm with a simple fitness function using robot relative localization

7.4 Final PSO algorithm for smoke source localization

This is the final step in the effort to implement a modified PSO algorithm for use with real quadrotor UAVs and the task of locating an odor source. All the previous algorithm versions are combined.

7.4.1 Simulation setup

This simulation has the following parameters:

- Robot dimensions taken into account
- Movement using the quadrotor dynamic model and regulator
- Fitness function based on smoke particles concentration
- PSO algorithm modified to avoid collisions of swarm members
- PSO algorithm modified for the specific application of locating a smoke source
- Continuous fitness function sensor reading every 0.3 seconds (not only at positions calculated by the algorithm)

The constants of the algorithm were tuned as follows:

parameter	value	units
w	0.2	–
c_1	1.4	–
c_2	1.6	–
c_3	3	–
$minDist$	140	cm
v_{max}	5	m
w_{max}	5	m
h_{max}	5	m

Table 4: Parameter values of the final PSO implementation

7.4.2 Simulation results

The progress of this simulation followed expectations. Due to the fact that only one quadrotor is allowed to travel while the other remain rested, the time duration of the algorithm's progress is significantly higher than the duration of the basic algorithm's progress. The number of iterations needed to achieve the goal remains roughly the same. The extended duration of the progress is also caused by the height limitation of newly calculated positions. It takes the swarm a long time to descend to the height of the smoke source. A view of the progress of the final algorithm is in figure 12.

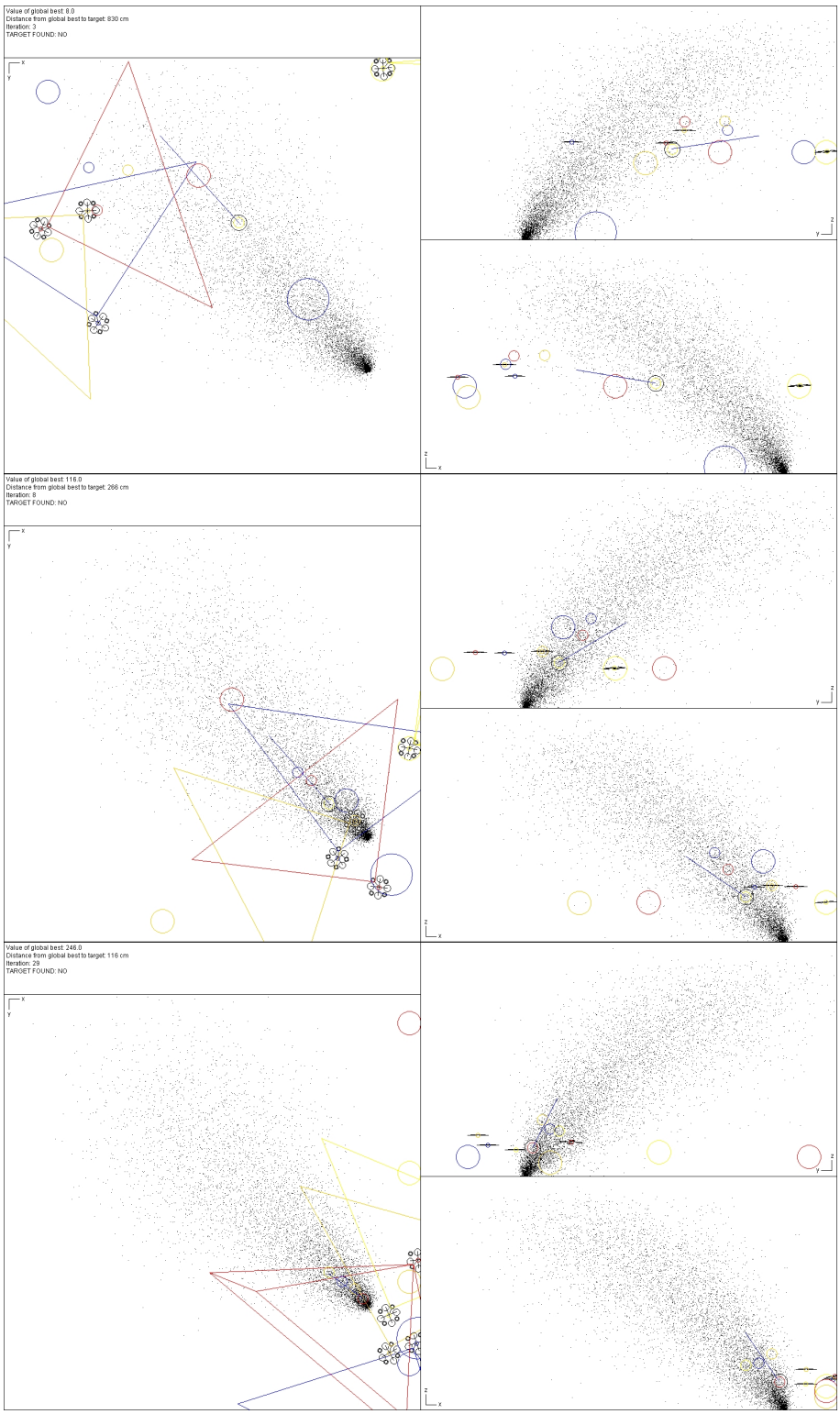

Figure 12: Final PSO algorithm for smoke source localization with collision avoidance

8 Conclusion

The aim of this work was to find out whether the basic version of the PSO algorithm can be modified for use with real autonomous robots and still be effective. After this was accomplished, the specific task of locating a source of an odor was selected for the algorithm's use. All the achieved results are hoped to be further improved and used in real life applications.

The basic version of the algorithm ignoring physical properties of robots converges quickly when the algorithm parameters are correctly tuned. When used for smoke source localization without any special modifications, the progress of the algorithm rarely ends in success due to the specific and changing nature of the fitness function. After the algorithm was modified using a mechanism for the prediction of the location of the global maximum, its efficiency rapidly increased, almost every time ending in a succesful localization of the source.

For use with quadrotor UAVs, the algorithm had to be modified to disallow swarm members to collide with each other. The presented approach led to a significant increase in time duration of the algorithm's progress while maintaining a similar number of iterations needed to reach the goal as in the basic PSO version.

The resulting algorithm developed in this work is functional and sufficient for its task, but may be improved using more sophisticated methods of collision avoidance and trajectory planning. Additionally, several factors that may improve the usability of this algorithm were not considered in this work. That includes the significant effect the propellers of the quadrotor have on the surrounding air, changes of smoke density due to turbulences etc.

Bibliography

[1] www.ifixit.com. 2013.

[2] Taeyoung Lee, Melvin Leok, and N. Harris McClamroch. Geometric Tracking Control of a Quadrotor UAV on SE(3). 2010.

[3] Jan Faigl, Tomáš Krajník, Jan Chudoba, and Martin Saska. Low-Cost Embedded System for Relative Localization in Robotic Swarms. 2012.

[4] Shuo Pang and Jay A. Farrell. Chemical Plume Source Localization. 2006.

[5] James Kennedy and Russell Eberhart. Particle Swarm Optimization. 1995.

9 Appendix

9.1 CD Contents

Directory name	Description
bp	bachelor thesis in pdf format.
project	NetBeans project of the entire simulation
experiments	pictures of the algorithms' progress

Table 5: CD contents