(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
8 December 2016 (08.12.2016)

WIPO | PCT

(10) International Publication Number
WO 2016/193667 A1

(51) International Patent Classification:
B64C 39/02 (2006.01)

(21) International Application Number:
PCT/GB2016/051427

(22) International Filing Date:
18 May 2016 (18.05.2016)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
1509510.2 1 June 2015 (01.06.2015) GB

(71) Applicant: **IMPERIAL INNOVATIONS LIMITED** [GB/GB]; 52 Princes Gate, Exhibition Road, London SW7 2PG (GB).

(72) Inventors: **KOVAC, Mirko**; Imperial College London, London SW7 2AZ (GB). **MCFARLANE, Edward**; Imperial College London, London SW7 2AZ (GB).

(74) Agents: **NOBLE, Nicholas** et al.; Kilburn & Strode LLP, 20 Red Lion Street, London WC1R 4PJ (GB).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*

WO 2016/193667 A1

(54) Title: AERIAL DEVICE CAPABLE OF ADDITIVE MANUFACTURING AND ASSOCIATED METHOD

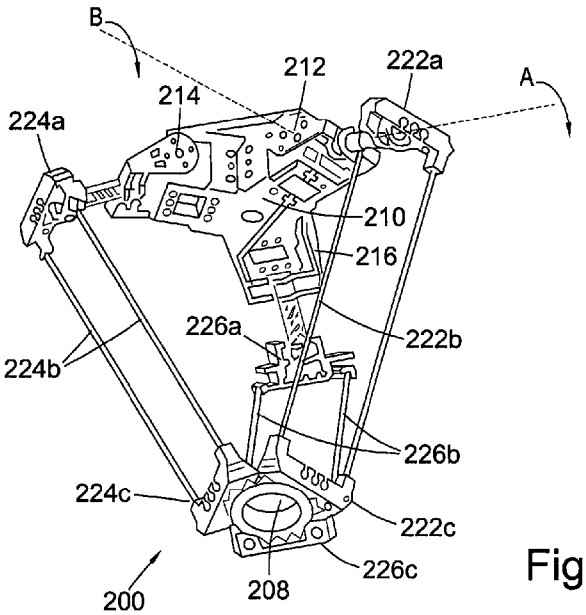


Fig. 2

(57) Abstract: An aerial device (100) capable of controlled flight comprises a first body (110) comprising a lift generator for providing lift to the aerial device (100); a second body (208) comprising a substance dispenser (122) for controllably dispensing a curable substance for additive manufacture; an articulated coupling assembly (200) coupling the first body (110) to the second body (208); an actuator (212; 214; 216) operable to articulate the articulated coupling assembly (200) so as to move the second body (208), and thereby the substance dispenser (122), relative to the first body (110); and a controller operable to control the lift generator, the substance dispenser (122), and the actuator (212; 214; 216).

AERIAL DEVICE CAPABLE OF ADDITIVE MANUFACTURING AND ASSOCIATED METHOD

Field

This disclosure relates to an aerial device capable of controlled flight, and in particular, but without limitation, to an aerial device comprising a manipulable substance dispenser for additive manufacturing.

Background

Robots are often used in the construction and architecture industries for fabrication purposes. Since the 1980s, mobile robots have been used to carry out assembly and construction tasks such as welding, painting, bricklaying and decommissioning. However, traditional ground construction robots such as gantry and industrial robots are often constrained by predefined working areas that hinder their scale of action, and that limit the size and scope of pieces of work they may act upon.

The use of flying robots for carrying out construction tasks is a solution that overcomes some of the main limitations of predefined working areas faced by ground robots. Some of the benefits that flying robots bring over traditional ground robots include:

- the ability to operate dynamically in space, especially in areas difficult to access, such as those at high altitude;
- the ability to work in co-operation with other flying robots to accelerate the construction process;
- the ability to more precisely lay out and assemble materials for construction;
- the ability to eliminate dangers for human workers, especially those involved in roof work or other high-altitude work, and the ability to safely navigate dangerous terrains.

Summary

Aspects and features of the invention are set out in the appended claims.

Brief description of the drawings

Examples of the present disclosure will now be explained with reference to the accompanying drawings in which:

Figure 1 shows a quadcopter;

Figure 2 shows a delta robot;

Figure 3 shows a block diagram of a system for implementing elements of the approach described herein;

Figure 4 is a sketch of aerial devices in various circumstances;

Figure 5 shows a model of a delta arm;

Figure 6 shows a build area map;

Figure 7 shows a delta arm prototype;

Figure 8 shows a charger prototype;

Figures 9 and 10 show CAD models of a delta arm;

Figure 11 shows position Z m data for head motion with dynamic simulation of motor torque 0.1 Nm and payload mass of 40g;

Figure 12 shows velocity Z ms^{-1} data for head motion with dynamic simulation of motor torque 0.1 Nm and payload mass of 40g;

Figure 13 shows a vertical male type charger port; and

Figure 14 shows a model of a delta arm.

Throughout the description and the drawings, like reference numerals refer to like parts.

Detailed description

An example of an aerial device is shown in Figure 1. The aerial device 100 comprises a number of rotors, which may for example be propellers, and which render the aerial device 100 capable of controlled flight. The aerial device 100 of Figure 1 has four propellers 120, and is accordingly referred to as a 'quadcopter'. The propellers 120 are mounted to a first body 110 of the aerial device 100, and are arranged to rotate relative thereto. When rotated, the propellers 120 provide lift to the aerial device 100, allowing it to fly. In the example of Figure 1, the body 110 includes arms 112 and rings 114.

The aerial device 100 further comprises a substance dispenser 122 operable to controllably dispense a curable substance for additive manufacture (i.e. for '3D printing'). In order to move the substance dispenser 122 relative to the first body 110 of the aerial device 100, the aerial device 100 comprises an articulated coupling assembly 200, which couples the first body 110 of the aerial device 100 to a second body of the aerial device 100 to which the substance dispenser 122 is coupled.

The articulated coupling assembly may form part of a so-called 'delta robot' or 'delta arm', as illustrated in Figure 2.

The delta robot 200 comprises two platforms: a base 210 and a second body (or head) 208. The base 210 of the delta robot 200 is fixedly mounted to the first body 110 of the aerial device 100, for example, to its underside. As another possibility, the base of the delta robot 200 may be integral with the first body 110 of the aerial device 100. The head 208 of the delta robot 200 is coupled to an end-effector - in this case the substance dispenser 122. As another possibility, the head of the delta robot 200 may be integral with the substance dispenser 122. The delta robot 200 enables the substance dispenser to be controllably moved in relation to a target site for dispensing the curable substance.

To this end, the delta robot 200 comprises three arms 222, 224, 226 coupling the base 210 to the head 208. Each arm 222, 224, 226 comprises a respective upper portion 222a, 224a, 226a a respective intermediate portion 222b, 224b, 236b and a respective lower portion 222c, 224c, 226c. Each intermediate portion comprises two parallel sub-arms, that are each coupled to the head 208 and the base 210 via respective ball and socket joint. Actuators 212, 214, 216 control the movement of arms 222, 224, 226 and at least a portion of each actuator 212, 214, 216 is mounted to the base 210 of the delta robot 210. The actuators 212, 214, 216 allow the upper portions 222a, 224a, 226a of the arms to rotate about two axes, thereby allowing the head 208 to rotate with two degrees of freedom. As an example, actuator 212 is operable to cause rotation of upper portion 222 about axes A and B of Figure 2.

Like most aerial devices, quadcopters suffer from small instabilities and drift in position, and maintaining a fixed reference point to the ground is challenging as disturbing forces, such as wind and ground effects, cannot be absorbed through a physical medium and instead must be compensated for through thrust vectoring. However, the inventors have appreciated that, as state of the art flight controllers can only achieve a hover accuracy of about 7cm while hovering close to the ground, their accuracy is not sufficient for 3D printing. The delta robot 200 is operable to decouple the movement of the base 210 and the head 208, thereby allowing the head 208 to maintain a constant position in spite of changes in the position of the base 210.

Figure 3 shows a block diagram of a system for operating the aerial device 100 as described herein. In particular, a controller 300 comprises a microprocessor 320 arranged to execute computer-readable instructions as may be provided to the

controller 300 via input/output means 322 which may be arranged, without limitation, to interface with one or more wired or wireless ports, for example a USB port. The microprocessor 320 may also store instructions in a memory 324, for example a random access memory. The microprocessor 320 is arranged to output results of executed programmes at the input/output means 322, and/or may communicate those results to another device via a network interface 328 that is arranged to couple, preferably in a wireless manner, the controller 300 to a communications network such as the internet (not shown in Figure 3). The microprocessor 320 may be further arranged to receive instructions and/or data via the network interface 328. In particular, the microprocessor 320 may receive commands from a base station which remotely controls the flight of the aerial device 100. The microprocessor may also be arranged to receive sensor data from one or more sensors 330, such as a camera or accelerometer.

As a result of executing the computer-readable instructions, the microprocessor 720 may be arranged to control the propellers 120 in order to fly the aerial device 100. In particular, the microprocessor 720 may control the angle of propellers 120 relative to the horizontal and/or the amount of lift provided by individual propellers 120. In addition, the microprocessor 720 may be arranged to control the substance dispenser 122, and/or to control the actuators 212, 214, 216 in order to move the head 208 of the delta robot 200 and thereby move the substance dispenser 122.

There are a number of ways in which the controller 300 can control the actuators 212, 214, 216 in order to move the head 208. For example, the controller may receive position-related information, and control the actuators 212, 214, 216 based on this information; for example, using the algorithm described in Section 3.2 below.

The position-related information may comprise image information from one or more cameras (which may form part of a motion capture or optic flow system), such as a Vicon, an Xbox Kinect®, Virtual Reality (VR) headset, and/or approaches such as those used in Project Tango, Steam Lighthouse, and Oculus Position Tracking. These cameras may be on-board the aerial device 100. Alternatively, the cameras may be on or near the surface on which the curable substance is to be dispensed, in which case the image information from these cameras could be received via the network interface 328.

The image information may enable the controller 300 to determine how far the substance dispenser 122 is from a target site, and to control the actuators 212, 214, 216 to move the substance dispenser 122 back towards the target site. This determination may be aided by placing markers (such as reflective markers) on the target site. Alternatively, cameras on or near the target site may determine the position of the substance dispenser 122 or aerial device, and may send a command to the controller 300 instructing it to change the position of the substance dispenser 122.

The position-related information may be supplemented by information provided by one or more inertial measurements units – for example from an accelerometer forming part of the aerial device 100.

As will be described in more detail below, the substance dispenser 122 is used to store, mix and dispense two or more chemicals that when combined produce an adhesive or other curable substance such as a foam.

The substance dispenser 122 comprises an actuator portion and a mixing portion. The actuator portion and mixing portion are each coupled to a pair of containers or reservoirs. Each reservoir contains a chemical stored therein.

The actuator portion comprises a lead screw driven by a motor. The lead screw is linked to plunger that is used to force the chemicals out of two reservoirs. The reservoirs are placed parallel and in-line with the lead screw. The power required for the motor is matched to the force required to move the plunger.

The mixing portion includes a Y-splitter fluidly linking the reservoirs to a mixing nozzle. The mixing nozzle is joined to a reaction chamber and a printing nozzle.

When activated by the motor, the lead screw causes the plunger to translate and force the two chemicals out of the reservoirs. The two chemicals are mixed by joining their respective flows with the Y-splitter and pumping them through a disposable mixing nozzle. The mixing nozzle is a static mixer, whose internal geometry enhances mixing of the two chemicals. At the exit of the mixing nozzle, the two chemicals are substantially fully mixed.

In order to increase the amount of time during which the two chemicals react, the two chemicals flow to the reaction chamber following their mixing in the mixing nozzle. In a preferred example, the mixture is of an adequate viscosity for deposition approximately 90 seconds after the flow initiates.

When foam or the desired adhesive begins to form in the reaction chamber, air is pressurised in the reaction chamber so that the material is pushed to the printing nozzle. The substance dispenser 122 includes a valve to prevent outflow of the chemical mixture prior to complete mixing and reaction.

A method of using the substance dispenser 122 will now be described. Under control of controller 300, the propellers 120 are activated and used to provide lift to the aerial device 100. The controller 300 further controls a camera module to collect image data of the surrounding environment. For example, using on-board cameras, the camera module is able to construct a map of the surrounding environment. Using techniques known in the art, the controller 300 may process the image data in order to controllably navigate or otherwise fly the aerial device 100 within the surrounding environment.

A target site in the surrounding environment is then identified. For example, the target site may be identified by a user transmitting the location of the target site to the aerial device 100. Alternatively, the target site may be identified by the controller 300 processing the image data generated by the camera module. The target site may be identified by a suitable computer vision software algorithm, for example by using SLAM (Simultaneous Localization and Mapping) or OpenCV (Open Source Computer Vision). Such software algorithms are able to create a 3D map of the surrounding environment based on the image data generated by the camera module.

Once the target site is identified, controller 300 controls propellers 120 so as to controllably fly the aerial device 100 towards the target site. In the present example, the aerial device 100 may fly or hover directly above the target site in order to be in a position to dispense the adhesive onto the target site. In other examples, however, it is envisaged that the aerial device 100 may be arranged to dispense the adhesive through other apertures; for example, the aerial device 100 may be arranged to dispense the adhesive laterally with respect to the aerial device 100, instead of

vertically downwards. In particular, the aerial device 100 may land adjacent the target site prior to applying the adhesive.

When in position above the target site, the controller 300 controls substance dispenser 122, and communicates with controlling circuitry to activate the actuator portion of the substance dispenser 300 and activate the plunger so as to begin the chemical mixing process. In some examples, the chemicals may be mixed before the aerial device 100 reaches the target site, such that the adhesive is ready for dispensing by the time the aerial device 100 has reached the target site. Once the controller 300 determines that the chemicals are mixed (for example once a predetermined amount of time has elapsed from the initiation of the mixing process), the valve of substance dispenser 122 is opened so as to allow the adhesive to be dispensed ejected via the printing nozzle.

In some examples, the aerial device 100 may be arranged to pick up objects and affix them to target sites using an adhesive delivered from substance dispenser 122. For example, it is envisaged that the aerial device 100 may include a gripping arm or similar object manipulator that may be arranged to manipulate or otherwise pick up objects from the environment.

The gripper may be under control of the controller 300. Thus, using the on-board imaging means or camera module, the controller 300 may control the flight of the aerial device 100 to position the aerial device 100 relative to the object to allow gripping or handling of the object. For example, using the on-board cameras, the controller 300 may process image data of the surrounding environment to identify an object relative to the aerial device 100. The aerial device 100 may then fly towards the identified object such that the gripper may then be in a position to grasp the object.

As one possibility, the controller 300 may be operable to detect an object such as one object that has been thrown or is falling- and to control the actuators so as to move the second body in order to catch the object therewith. Detection of the object may be performed by the controller by way of processing the image information.

As one possibility, the controller 300 is operable to control the actuators in order to soften a landing on a surface by the second body. When the aerial device 100 has its second body or end effector in contact with a surface – as may occur, for example, following such a landing, the controller 300 may be operable to control the actuators in order to rapidly exert a force on the surface so as cause the aerial device to jump relative to the surface.

Figure 4 shows a number of examples of an aerial device capable of controlled flight. In the examples of Figure 4, the second body is part of a flexible trunk that is flexible relative to the main body of the flying platform of the aerial device and has a nozzle as its substance dispenser at an end thereof. The nozzle may be accompanied by a sharp tip to enable the flexible trunk to perforate substances – such an in order to inject biological material into soil. The flexible trunk may further be operable, in a serpentine manner, to entwine itself around surfaces – such as that of a plant or other material (for example so as to enable the aerial device to pick up material that would be hazardous to a human). Further, the flexible trunk may be arranged to enable the aerial device to hang on to, or perch on, an object and/or to hang therefrom.

In Figure 4, reference signs 401 to 414 denote:

- 401: Flying platform, could be extended with legs or wheels to move on ground or walls;
- 402: Storage of chemicals or biological material (fertilizer, glue, bacteria, fungi);
- 403: Flexible "trunk" used to spray materials and manipulate objects;
- 404: Sharp tip and nozzle;
- 405: Landing on ground and injecting biological material in the soil to manage soil properties;
- 406: Trunk is penetrating soil; similar to a plant root;
- 407: Nozzle is spraying organic material or chemicals;

- 408: Soil;
- 409: Manipulating objects with flexible trunk that adapts to object geometry;
- 410: For example, weed;
- 411: For example, hazardous material;
- 412: Perching to objects using trunk to attach to structures and observe environment;
- 413: For example, tree or industrial structure; and
- 414: Observe/inspect environment.

While the above has been described with respect to an articulated coupling assembly forming part of a delta robot, other articulated coupling assemblies could be used, such as an XYZ stage, an XY gimbal with Z radial depth, a three-axis articulated robotic arm, and/or any of the examples set out in the below table.

Arm Type	Description	DOF(s)
Articulated	Robotic arms facilitate movement similar to Human arms. Actuated joints connect to form a kinematic chain.	1-7
Gimbal	Pivoted support that allows rotation around a single axis. Combined to provide support for multiple axis. Used in camera rigs for stabilisation.	1-3
Linear	Linear stages precisely control motion on an axis. Stages are added to incorporate the DOF(s) required with 3D printers typically taking this form.	1-3
Delta	Parallel type robot where translation is controlled through triangulation of the head. Optional linkage for rotations provide 4 DOF. Used in industrial food processing.	3-4
Combination	Application of multiple types.	-

Furthermore, while in the above a substance dispenser has been used as the end-effector, it will be appreciated by those skilled in the art that the head 208 of the aerial device 100 could instead hold a different end-effector, such as a drill or any other mechanical tool.

There is described herein an aerial device which can be easily controlled and positioned precisely.

There is described herein an aerial device with a head which is lightweight and can thus be moved with low energy and high acceleration, due to the actuators being fixedly coupled to the base (and not the head).

There is described herein an aerial device which is able to maintain a steady absolute position, rather than only a steady look direction.

The approaches described herein may be embodied on a computer-readable medium, which may be a non-transitory computer-readable medium. The computer-readable medium carrying computer-readable instructions arranged for execution upon a processor so as to make the processor carry out any or all of the methods described herein.

The term "computer-readable medium" as used herein refers to any medium that stores data and/or instructions for causing a processor to operate in a specific manner. Such storage medium may comprise non-volatile media and/or volatile media. Non-volatile media may include, for example, optical or magnetic disks. Volatile media may include dynamic memory. Exemplary forms of storage medium include, a floppy disk, a flexible disk, a hard disk, a solid state drive, a magnetic tape, or any other magnetic data storage medium, a CD-ROM, any other optical data storage medium, any physical medium with one or more patterns of holes, a RAM, a PROM, an EPROM, a FLASH-EPROM, NVRAM, and any other memory chip or cartridge.

As will be appreciated by those skilled in the art, the components of the aerial device can be produced via additive manufacturing, for example via the use of a 3D printer. First, a computer-readable file containing data representative of an aerial device is produced. The data may be representative of the geometry of successive cross-sections of the component. This data is often called 'slice' or 'layer' data. The data can be produced from a Computer Aided Design (CAD) file, or via the use of a 3D scanner. A 3D printer can then successively lay down layers of material in accordance with the cross-section data to produce the aerial device components.

Further applications of the aerial device described herein include use for repair, inspection, manipulation or chemical treatment of a surface. Also the aerial device could act as a surgical device on animals, plants or people or perform medical assessment or treatment.

There will now be described several further examples relating to aerial devices capable of controlled flight. Features of the examples described below may be readily combined with features described above, as would be understood by the skilled person.

Overview

This disclosure describes the creation of utilities for the enhancement of micro aerial vehicle, MAV, in the field of aerial construction. Delta arm parallel robot was proposed to provide desired properties for MAV systems. A geometric solution to the delta arm triangulation problem was developed and integrated to the controls system. Creation of a delta arm prototype was made which includes electronic components and control systems that allow for integration onto drone architectures. Finally, the scope of further work to be completed has been outlined for the project.

1 Introduction

Aerial robots allow for greater versatility in a wide range of dynamic environments. Construction with the use of aerial robotics is a highly active research area with the promise of developing drones for rapid and advanced construction of complex structures. Micro aerial vehicles, MAVs, use advanced control algorithms to achieve stable flight.

Improving the stability of MAV based actuators would be beneficial for a plethora of MAV based applications. Decoupling an actuator from the MAV to form a two bodied system would allow the actuator to react to errors in the drone's position and provide greater manoeuvrability in the actuator's position.

1.1 Aims and Objectives

The aim of this project is to develop a full prototype robotic system to be used on UAVs and the necessary support technologies required for aerial construction. The prototype will be tested in its ability to provide enhancements in manipulating and operating on ground based systems. This would allow for mathematical model to be

devised for use on aerial based platforms and act as a proof of concept for future devices.

2 Background

Robotics is continuously developing new ways to interact with the environment. A wide variety of robots utilise different arm like configurations to provide the desired degrees of freedom, DOF. The more DOF required the greater complexity introduce and is therefore usually limited to design specific applications. Camera stabilisation is a common problem for MAVs, which is achieved through gimbal mechanisms that control the cameras rotation. Grippers on aerial vehicles are common in commercial application of helicopter deforestation. Actuators on MAVs include grippers mounted directly to the quad copter and full 6 DOF robotic arms have been demonstrated.

Control for MAVs is highly important. Contact or forces imparted to the drone are important to understand for countering their negative effects. Advanced control schemes have been devised for MAV flight that are able to interact with objects.

Recently, construction with MAVs has been implemented and their form and functions have been studied. There are different approaches to the type of structures and the level of prefabrication required. Many structures developed with drones are temporary design implementations for non critical applications. Accuracy of 3D deposition limits the unit size of block material used.

3 Theory

In this section, the mathematical models and computer theory are developed for solving the controls of the system and development of electronic systems. Robotic arms require a mathematical solution to relate the position to arm angles. From this a model for control is developed which allows desired positions to be translated to motor rotations. The algorithm was developed to create visualised solutions for both proof of concept and verification. Later models were ported to the control system to allow for calculation on the independent system.

3.1 Stability Techniques

Quadcopters suffer from small instabilities and drift in position, similar to all flying vehicles. Maintaining a fixed reference point to the ground is challenging as forces acting in disturbance, such as wind and ground effects, cannot be absorbed through a physical medium and instead must be accounted through thrust vectoring. Quadcopters control is inherently reactive to disturbance, with state estimation only

being able to predict a position within an error margin. Latency is also an issue with fixed pitch quadcopters reacting slower, than variable pitch, due to inertia of the blades. Therefore, interaction between bodies must be error tolerant. Creating a stabilising two-body system involves separating the desired degrees of freedom from the drone platform. Gimbals for cameras are currently used to correct for motion in flight and remove vibrations by controlling the rotations. For construction, position is required to be constraint to provide a static reference frame with respect to the world.

To separate the drone and device, a decoupled system was devised. The decoupled system was implemented through a controlled robotic arm for which various types were investigated. Solutions that provide high acceleration with low inertia are desired. Explored systems included an XY Z stage, XY gimbal with Z radial depth(with respect to rotated frame), 3-axis articulated robotic arm and delta arm. Delta arm was chosen for its properties of low inertia linkages, which reduces force imparted to the drone, and high accelerations that can be achieved.

3.2 Delta Arm Solver

A delta robot is a type of parallel robot. It houses motors in the base with upper arm linkages restricted to one plane of rotation. Lower arm linkages form a parallel connection that restricts rotations of the head of the robot. Three arms provide control over the desired translation. Motor rotation are measured from the XY plane in-line with the base

Actuators lie in the base of the design and are arrayed around the centre point, z-axis, to provide an equispaced 120°. Three arms are the minimum required to restrict all the axis of motion. Table 3.1 describes the parameters that define the delta arm model.

TABLE 3.1: Delta Arm Parameters

Parameter	Description
l_u	Upper arm length (m)
l_l	Lower arm length (m)
r_b	Radial distance from base centre (m)
r_h	Radial distance from head centre (m)
z_{nil}	Head zero position (m)

Forward solution to the desired position vector $\mathbf{x}^T = [x, y, z]$ for the joint angles, $\mathbf{q}^T = [\theta 1, \theta_2, \theta_3]$, is solved to provide "end-effector" (head of the delta arm) control. Normalised vectors are define for three points

$$\mathbf{v_1} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \mathbf{v_2} = \begin{bmatrix} -0.5 \\ \frac{\sqrt{3}}{2} \\ 0 \end{bmatrix}, \mathbf{v_3} = \begin{bmatrix} -0.5 \\ -\frac{\sqrt{3}}{2} \\ 0 \end{bmatrix} \tag{3.1}$$

with vectors being defined in a radial array on the XY plane with v1, v2, and v3 at angles to the X axis of $\theta = 0, 120, 240$ respectively (defined in a counter-clockwise direction). The base and head are offset from the origin to accommodate the motors and device. Head, h_n, and base, b_n, nodes are found by

$$\mathbf{h_n} = r_h \mathbf{v_n} \tag{3.2}$$

$$\mathbf{b_n} = r_b \mathbf{v_n} + \mathbf{P} \tag{3.3}$$

where $\mathbf{P}$ represents the desired position vector with respect to the origin at the base of the quad copter. Planes are defined, by vector notation, for each upper arm which defines the plane the arm lies in.

$$\mathbf{p_1} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \mathbf{p_2} = \begin{bmatrix} \frac{\sqrt{3}}{2} \\ 0.5 \\ 0 \end{bmatrix}, \mathbf{p_3} = \begin{bmatrix} -\frac{\sqrt{3}}{2} \\ 0.5 \\ 0 \end{bmatrix} \tag{3.4}$$

Solution is calculated through the intersection of a circle and sphere in 3D space. The circle represents the possible locations of the upper arm rotating in 1 DOF, around the base and the sphere, the lower arm rotating around the head in 2 DOF. Three solutions exists that give 0, 2 or infinite number of possible points. No solution is found if the total length is less than the distance of the head to the base, 2 solutions are found for a well formed problem, and infinite if the circle lies on the sphere.

$$f(n) = \begin{cases} 0 & \text{if } l_u + l_l < |\mathbf{h_n} - \mathbf{b_n}| \\ 2 & \text{if } l_u + l_l > |\mathbf{h_n} - \mathbf{b_n}| \\ \infty & \text{if } (\sqrt{l_l^2 - l_u^2} == \mathbf{h_n} - \mathbf{b_n}) \text{ and } (\mathbf{h_n} \times \mathbf{b_n} == 0) \end{cases} \tag{3.5}$$

The vector defining the circle to the sphere

$$\mathbf{cs_n} = \mathbf{h_n} - \mathbf{b_n}. \tag{3.6}$$

Defining a projection point, sm, that projects the sphere point onto the plane as

$$\mathbf{sm_n} = (\mathbf{p_n} \cdot \mathbf{cs_n})\mathbf{p_n}. \tag{3.7}$$

The circle to projection point is defined as

$$\mathbf{cm_n} = \mathbf{cs_n} - \mathbf{sm_n}. \tag{3.8}$$

Radius sphere projection, s, is given as

$$s = \sqrt{l_l^2 - |\mathbf{sm_n}|^2} \tag{3.9}$$

and defines, by use of Pythagoras theorem, the distance the spheres interaction point makes with the projected point sm. If $s = 0$ both solutions will be identical.

Transformation between the defined plane is given through transformation vectors

$$\mathbf{i} = \frac{\mathbf{cm}}{|\mathbf{cm}|} \tag{3.10}$$

$$\mathbf{j} = \mathbf{p_n} \times \mathbf{i} \tag{3.11}$$

and the points found in space to be

$$\mathbf{x} = \frac{|\mathbf{cm}|^2 - s^2 + l_u^2}{2|\mathbf{cm}|} \tag{3.12}$$

$$\mathbf{y} = \sqrt{l_u^2 - x^2}. \tag{3.13}$$

5

The two vector positions can then be translated back to Cartesian space with two solutions given as

$$\mathbf{A} = \mathbf{b_n} + i\mathbf{x} + j\mathbf{y} \tag{3.14}$$

$$\mathbf{B} = \mathbf{b_n} + i\mathbf{x} - j\mathbf{y}. \tag{3.15}$$

In order to select the correct solution, constraints are applied. If *A* or *B* contains an imaginary part the solution is non physical, unreachable. If both solutions are real the longest vector formed from the head to origin is the correct point. Pythagoras theorem can then be used to calculate the angle that the upper arm forms with the solved point. Each arm acts independently, and can therefore be updated independently, however the system remains coupled.

3.3 Delta Arm Model

Parameters of the delta arm are required to satisfy the necessary constraints of the drone shown in Table 4.1. Physical limitations in the build area can be visualised from taking into account the delta arm parameters, Table 3.1, which define sizing and the joint limitations for both the upper and lower arms. Rotations for the upper arm were taken between 0° – −90° to allow for quad copter mounting. Lower arm rotations occur on two separate joints, both predicted to act between the range of −45° – +45°. Figure 5 shows the delta arm parameters with corresponding build area with limitations shown in Figure 6. Sizing was based on build area simulations and available materials.

Delta arm dynamics are coupled to the motion of the quad copter and positioning of the arms. A simple rigid model was created to solve the initial forces parted to the drone. The lower arm linkages are presumed to act as a single node. A predicted payload of 40g was prescribed for the payload head. In extreme manoeuvres quad copters can move at 3.5ms^{-1} with accelerations of up to 15ms^{-1}. Under hover conditions acceleration can be assumed to be much lower.

Figure 5:

Delta arm matlab model showing sizing and positioning of joints for $l_u = 0.06, l_l = 0.15, r_h = 0.04, r_b = 0.02$ and $z_{nil} = 0.16$.

Figure 6:

Build area map, iterative points scheme for $l_u = 0.06, l_l = 0.15, r_h = 0.04, r_b = 0.02$ and $z_{nil} = 0.16$.

Torques required by the motor can be predicted through solving a simple ridged body model. Lower arms are modelled as a single strut. Internal stresses are ignored. Holding torque can be modelled with the delta arm at 0°, the default position. Torques can then be calculated by

$$T_H = Fd$$

$$T_H = (Payload(g + a))/3l_u$$

$$T_H = (0.04(9.81 + 15))/30.06 = 0.02Nm$$

which can be used to estimate the motors required. As the lower arm angle goes to 0° (with respect to XY plane) the internal stresses in the arms will increase. This is important for validating the build area to ensure the response can be uniform across the device.

3.4 Kinematics

Kinematics is a branch of classical mathematics that is used to describe the motion of bodies based on the derivatives, satisfying the second law of motion. Inverse kinematics refer to the use of kinematic equations to describe the relation between the "end-effector" and the system, an analytical solution to the geometric argument presented above. Inverse kinematic play a key part in robotic control theory. Systems are modelled as a set of linked nodes, known as a kinematic chain, each with a parent and a child node. A desired nodal position, usually the end-effector, can be set with solutions obtained for the rotation space.

Kinematic equations can be formed into the Jacobian form by taking the derivatives. The Jacobian form is defined as

$$J = \frac{\partial x}{\partial q}, J = \frac{\partial x}{\partial t}\frac{\partial t}{\partial q}, \dot{x} = J\dot{q} \tag{3.16}$$

where the velocity vector in Cartesian space, $\dot{x}^T = [dx, dy, dz, \omega_x, \omega_y, \omega_z]$, includes rotations and q defines joint angles. Equation 3.17 defines the mass matrix for each joint, i, with corresponding mass and inertia. Jacobian is formed around the centre of mass of each nodal link.

$$M_x = \begin{pmatrix} m_i & 0 & 0 & 0 & 0 & 0 \\ 0 & m_i & 0 & 0 & 0 & 0 \\ 0 & 0 & m_i & 0 & 0 & 0 \\ 0 & 0 & 0 I_{xx} & I_{xy} & I_{xz} & \\ 0 & 0 & 0 I_{yx} & I_{yy} & I_{yz} & \\ 0 & 0 & 0 I_{zx} & I_{zy} & I_{zz} & \end{pmatrix} \tag{3.17}$$

Conservation of kinetic energy can be applied to the Jacobian to transfer between Cartesian and joint space.

$$KE = \frac{1}{2}\sum_{i=0}^{n}(\dot{x}_i^T M_x \dot{x}_i) \tag{3.18}$$

Substituting 3.16 into 3.18 gives

$$KE = \frac{1}{2}\dot{q}^T \sum_{i=0}^{n}(J_i^T M_x J_i)\dot{q}$$

$$M_q = J^T M_x J \tag{3.19}$$

Acceleration of the system in joint space:

$$\ddot{q} = (\tau - b(q, \dot{q}) - \tau_g)/M_q \tag{3.20}$$

where $\tau$is the torque applied, b is a function defining the Coriolis and centrifugal effects and τ_g is the force applied due to gravity.

$$\tau_g = \sum_{i=0}^{n}(J_i^T F_{gi}) \tag{3.21}$$

4 Design and Implementation

This section introduces the design choices and implementation of the robotic arm and construction enabling technologies. Design choices are chosen under constraints of the drone platforms, budget and manufacturing tools.

TABLE 4.1: Drone Constraints

Drone	Size (mm)	Payload (g)	Accuracy (±cm)
Hummingbird	540 x 540 x 85.5	200	2
Pelican	651 x 651 x 188	650	2

4.1 Delta Arm

The delta arm robot is required to be light and fast to facilitate its design goal of being mounted to a drone. Manufacturing of the prototype was done using the available facilities under the Aerial Robotics Lab and Aeronautical Engineering workshop at Imperial College London.

A CAD simulation was developed to enhance development of the robotic arm (shown in Figures 9 and 10). Simulations were run to estimate real world performance and predict behaviour(A.3, A.4).

Polyoxymethylene, POM, also known as Acetal and Delrin was the main material used to produce the parts for the Delta Arm. POM is an engineering thermoplastic used in precision parts requiring high stiffness, low friction and excellent dimensional stability. Acrylic was also considered but the higher friction and brittle nature proved less suitable. 3D printing from a Connex system was available but deemed too expensive with material properties inferior to that of POM. Carbon fibre rods were used in manufacturing the linkages and supporting rods for its high stiffness and low weight properties.

Parts were manufactured with a Laser cutter to produce the required accuracy. Accuracy of cutting proved to be around ±0.2mm either side of the design pattern, measured with micrometer, with variations due to the engravement path. Laser parts were assembled from 2mm POM with a jigsaw style like configuration to allow for a more rigid assembly. Helper parts were created to position pieces for gluing and enforce structure alignment, which increased accuracy and lowered friction of the final assembly.

Lower arm joints have 2 DOF to create the parallelogram type structure. Commonly found in other and larger delta arm robots are ball joints which provide the required

DOF with one joint. We were unable to source appropriate sized rod end joints, and therefore created a 2 joint hinge.

TABLE 4.2: Delta Arm Mass Properties

Part	Simulated (g)	Prototype (g)
Head	5	6
Base	72	74
Upper Arm	5	5
Lower Arm	2	3
Total	109	120

5

Figure 7: Built delta arm prototype shown at $\theta_i = 0°$ position.

4.2 Controller

Control of the robotic arm needs to receive and command the delta arm with minimal latency. Commands can be received in the form of an offset position compute triangulation of the arm, offloading data processing from computer controller, receive wireless commands from the drone, either through link to drone or wireless link to drone controller. Faster processors will reduce latency as computational time decreases.

The ARM mbed NXP LPC1768 microcontroller is a Cortex-M3 processor in a rapid prototyping form for use in designing different systems. Cortex-M3 has a high clock speed and should prove adequate to controlling the prototype.

Wireless communication is provided by the Xbee WiFi which provides a wireless N network for high data transmission and range. WiFi was chosen over bluetooth and radio to allow for a full digital system with high payloads and ease of connection to controller devices over a wireless network or ad-hoc system.

Communication between devices and controller is formatted into message packets using Google Protobuf. Protocol Buffers is a system developed by Google for communication of serialised data structures that is platform and language neutral. Communication data structures are written once and compiled for different frameworks providing content type protection. This allows communication between a wide variety of devices. Message structure, communicate commands and data positions with error responses.

4.3 Motors

Actuator devices for the drones arms are required to be light and powerful enough to provide the necessary power and weight to be utilised on a drone architecture. Motor step and position detection are important to maintain accuracy in movements. It was proven difficult to produce a motor to provide necessary accuracy with weight restrictions for budget stated. We chose the XL-320 above custom encoding devices and brushless motors due to weight, cost and control.

TABLE 4.3: Dynamixel XL-320 Properties

Weight (g)	Dimensions (mm)	Resolution (o)	Stall Torque (Nm)
19	24 x 36 x 27	0.29	0.39

Largest error in position is given when the upper and lower arm of the delta robot become perpendicular with respect to the plane. Small angle approximation the position error, e, can be evaluated for a rotation error, θ_e, as

$$\mathbf{e} = \pm l_u sin(\theta_e) \frac{\mathbf{j} - \mathbf{h}}{|\mathbf{j} - \mathbf{h}|}$$

where j and h represent the joint and head positions. Maximum position displacement is directly related to θ_e as $l_u sin(\theta_e)$.

XL-320 is digitally controlled servo motor. The motor driver(A.1 A.2) was written for the mbed platform following the Dynamixel communication protocol 2.0. The motor features an integrated PID controller which can be set by the controller. This is integrated into the wireless communication to allow the driver to tune the PID in flight which would allow for dynamic operations of payload mass.

4.4 Charging

Aerial robotics is largely limited by current generation battery technology. Batteries are relatively heavy with optimal payload versus flight time leaving much to be desired. Manually recharging drones with human interaction is a tedious problem that stops drones from becoming fully autonomous vehicles. A reliable mechanism to connect and recharge the energy stores (or batteries) of drones safely would be a significant step in achieving autonomy.

Lithium-ion polymer batteries, LiPo batteries, is the current technology used in most quad copter drones. Voltage depends on its chemistry and the number of cells used, typically around 3.7V per cell. Batteries are charged with the use of specialised LiPo battery charges that connect with two sets of wires to the battery; discharge leads and balance leads. Balance leads provide connection to each cell. Balancing of the battery cells is required to ensure the battery is charged safely.

From a technology point of view, a simple physical connection system was devised over induction and other more complex solutions, such as laser power transfer, due to constraints in technology maturity and added weight and complexity to the drone. Induction has no physical connection with the charging base which provides a near

field of electromagnetic resonance. The charging controller would be required to be built onto the drone, to provide balancing of the battery, which increases weight and complexity. Physical contacts propose a problem due to controller stability especially under ground effect conditions. An error tolerant connection device was required.

Two concepts were devised for drone mounted charging, shown in Figure 8 and 13. Both are similar in operation with different layout of pins in a flat head style configuration, Figure 8, and a key and barrel lock style, Figure 13. The physical connection comes from spring loaded contacts, known as pogo pins, that act as conductive spring connectors. The flat head configuration, orientation and position are controlled. Tolerance is provided through a large chamfered edge that resembles the error margin of the drone, acting as a funnel for the charging port. Magnets are placed on either side of the port and base, with opposing polarity to ensure correct orientation and opposing polar orientation on the port to ensure firm connection. Mounting of the port would be on the base or arm linkage of the drone. Vertical configuration, Figure 13, relinquishes orientation by forming a vertical column of connectors.

Base mechanism proposed is passive and depends on the accuracy of the drone. Base mounting could be spring based to reduce force imparted to the drone on docking. Manufacturing and testing is required to prove validity of devices.

Figure 8: Charger prototype features pogo pin and magnetic connectors.

5 Further Work

In this section we include some further work and highlight the importance of the ongoing developments with explanations of how they will be accomplished.

5.1 Drone Integration

Mounting of the prototype delta arm to the Hummingbird and Pelican will be a significant goal in this project. Controller is tested and all basic functionality for control was achieved. Three main methods of control for the platform are foreseen. Stability: counteracting drone position drift, active: fast actuations superimposed on the drones reference position, and dynamic: dynamic modelling of the system to perform dynamic interactions.

5.2 Application Prototypes

Prototype delta arm robot has been developed. Further development would incorporate additional features such as a foam dispensing mechanism that will allow for the creation of 3D structures. A foam-dispensing mechanism has already been developed and it would be used in conjunction to provide a set of useful tools showing the full potential of the device.

Development of other simple prototype applications will be able to showcase the different control types. A simple wireless camera with operator control, gripper and gyro measured payload device, could display the different control types and useful applications for the delta arm.

5.3 Dynamic Modelling

A full dynamic model would allow for control state estimation to be integrated into the controller, which potentially allow for greater stability in flight. Further dynamic simulations could be experimented with to provide dynamic operations, for instance the delta arm could be modelled as a spring and damper system between the quadcopter and payload reducing impact forces from rapid manoeuvres.

5.4 Charging

Charging requires testing of conceptual ideas to form a reliable docking mechanism. Tolerance and charge rate along with weight are important to document and analyse. PCB has been constructed to form plates for connections along with 3D printed versions of the CAD models. Finally, mounting of charging devices to quad copters for extend anonymous flight would prove a significant goal in this project.

A. Appendix

Figure 9: Dynamic cad model used to simulate forces on model.

Figure 10: Dynamic cad model used to simulate forces on model.

Figure 11: Position Z m data for head motion with dynamic simulation of motor torque 0.1 Nm and payload mass of 40g

Figure 12: Velocity Z ms^{-1} data for head motion with dynamic simulation of motor torque 0.1 Nm and payload mass of 40g

Figure 13: Vertical male type charger port, for key lock style charging station.

```
1  /* Dynamixel Communication 2.0 */
2  /* Implements the dynamixel communication protocol for XL-320. */
3  #ifndef MOTORS_H
4  #define MOTORS_H
5
6  #include "mbed.h"
7
8  class DM2 {
9
10 public:
11         // Create Dynamixel Communication protocol 2.0
12     DM2(PinName tx, PinName rx, int baud=1000000);
13
14     int Test(int ID=1);

15      int Ping(int ID=1);
16      int SetID(int ID, int newID);
17      int SetBaud(int ID, int rate);
18      int SetLED(int ID, int colour);
19      int Rainbow(int ID);
20      int SetP(int ID, int value);
21          int SetI(int ID, int value);
22          int SetD(int ID, int value);
23          int SetGoalPosition(int ID, int angle);
24          int SetPunch(int ID, int punch);
25
26  private :
```

```
27        // REPLY BUFFER
28      static unsigned char recycle[255];
29
30        int statusError(unsigned char* buf);
31        int length(unsigned char* buf);
32        int validate(unsigned char* buf);
33        void flush();
34        void packetPrint(int bytes, unsigned char* buf);
35
36        void write(unsigned char* buf, int n);
37        int read(int ID, unsigned char* buf, int nMax=255);
38        int send(int ID, int bytes, unsigned char* data, unsigned char ins=0x03, unsigned char* reply=recycle);
39        //int ping(int ID);
40        int dataPack(int start, unsigned char* data, int address, int value);
41        int dataPush(int ID, int address, int value);
42
43      Serial _out;
44      Serial _in;
45      int _baud;
46      int _bitPeriod;
47  };
48
49  // EEPROM
50  #define XL_MODEL_NUMBER_L 0
51  #define XL_MODEL_NUMBER_H 1
```

```
#define XL_VERSION 2
#define XL_ID 3
#define XL_BAUD_RATE 4
#define XL_RETURN_DELAY_TIME 5
#define XL_CW_ANGLE_LIMIT_L 6
#define XL_CW_ANGLE_LIMIT_H 7
#define XL_CCW_ANGLE_LIMIT_L 8
#define XL_CCW_ANGLE_LIMIT_H 9
#define XL_CONTROL_MODE 11
#define XL_LIMIT_TEMPERATURE 12
#define XL_DOWN_LIMIT_VOLTAGE 13
#define XL_UP_LIMIT_VOLTAGE 14
#define XL_MAX_TORQUE_L 15
#define XL_MAX_TORQUE_H 16
#define XL_RETURN_LEVEL 17
#define XL_ALARM_SHUTDOWN 18
// RAM
#define XL_TORQUE_ENABLE 24
#define XL_LED 25
#define XL_D_GAIN 27
#define XL_I_GAIN 28
#define XL_P_GAIN 29
#define XL_GOAL_POSITION_L 30
#define XL_GOAL_SPEED_L 32
#define XL_GOAL_TORQUE 35
#define XL_PRESENT_POSITION 37
#define XL_PRESENT_SPEED 39
#define XL_PRESENT_LOAD 41
#define XL_PRESENT_VOLTAGE 45
#define XL_PRESENT_TEMPERATURE 46
#define XL_REGISTERED_INSTRUCTION 47
#define XL_MOVING 49
#define XL_HARDWARE_ERROR 50
#define XL_PUNCH 51
// INS
const unsigned char INS_Ping = 0x01;        // Corresponding device ID command to check if packet reaches
const unsigned char INS_Read = 0x02;        // Read command
const unsigned char INS_Write = 0x03;       // Write command
const unsigned char INS_RegWrite = 0x04;    // When receiving a write command packet data is not immediatel
const unsigned char INS_Action = 0x05;      // Go command for Reg Write
const unsigned char INS_Factory = 0x06;     // Reset All data to factory default settings
const unsigned char INS_Reboot = 0x08;      // Reboot device
const unsigned char INS_StatusReturn = 0x55; // Instruction Packet response
const unsigned char INS_SyncRead = 0x82;    // Read data from the same location and same size for multiple de
```

```
96  const unsigned char INS_SyncWrite = 0x83;   // Write data from the same location and same size for multiple de
97  const unsigned char INS_BulkRead = 0x92;    // Read data from the different locations and different sizes for mu
98  const unsigned char INS_BulkWrite = 0x93;   // Write data from the different locations and different sizes for mu
99  // ID
100 const unsigned char ID_Broadcast = 0xFE; // 254(0xFE) is used as the Broadcast ID
101 // Util
102 #define DM_MAKEWORD(a, b) ((unsigned short)(((unsigned char)(((unsigned long)(a)) & 0xff)) | ((unsigned
103 #define DM_LOBYTE(w) ((unsigned char)(((unsigned long)(w)) & 0xff))
104 #define DM_HIBYTE(w) ((unsigned char)((((unsigned long)(w)) >> 8) & 0xff))
105
106 unsigned short update_crc(unsigned short crc_accum, unsigned char *data_blk_ptr, unsigned short data_blk_size)
107
108 #endif
```

LISTING A.1: Motor driver for XL-320 (motors.h)

```
#include "motors.h"
#include "mbed.h"
#include <math.h>

DM2::DM2(PinName tx, PinName rx, int baud)
        : _out(tx, NC)
        , _in(NC, rx)
{
        _baud = baud;
        _bitPeriod = 1.0/_baud;

        _out.baud(_baud);
        _in.baud(_baud);
}

unsigned char DM2::recycle[255] = {0};

// Dynamixel Communication 2.0 Checksum
unsigned short update_crc(unsigned short crc_accum, unsigned char *data_blk_ptr, unsigned short data_blk_size)
    unsigned short i, j;
    unsigned short crc_table[256] = {
        0x0000, 0x8005, 0x800F, 0x000A, 0x801B, 0x001E, 0x0014, 0x8011,
        0x8033, 0x0036, 0x003C, 0x8039, 0x0028, 0x802D, 0x8027, 0x0022,
        0x8063, 0x0066, 0x006C, 0x8069, 0x0078, 0x807D, 0x8077, 0x0072,
        0x0050, 0x8055, 0x805F, 0x005A, 0x804B, 0x004E, 0x0044, 0x8041,
        0x80C3, 0x00C6, 0x00CC, 0x80C9, 0x00D8, 0x80DD, 0x80D7, 0x00D2,
        0x00F0, 0x80F5, 0x80FF, 0x00FA, 0x80EB, 0x00EE, 0x00E4, 0x80E1,
        0x00A0, 0x80A5, 0x80AF, 0x00AA, 0x80BB, 0x00BE, 0x00B4, 0x80B1,
```

```
        0x8093, 0x0096, 0x009C, 0x8099, 0x0088, 0x808D, 0x8087, 0x0082,
        0x8183, 0x0186, 0x018C, 0x8189, 0x0198, 0x819D, 0x8197, 0x0192,
        0x01B0, 0x81B5, 0x81BF, 0x01BA, 0x81AB, 0x01AE, 0x01A4, 0x81A1,
        0x01E0, 0x81E5, 0x81EF, 0x01EA, 0x81FB, 0x01FE, 0x01F4, 0x81F1,
        0x81D3, 0x01D6, 0x01DC, 0x81D9, 0x01C8, 0x81CD, 0x81C7, 0x01C2,
        0x0140, 0x8145, 0x814F, 0x014A, 0x815B, 0x015E, 0x0154, 0x8151,
        0x8173, 0x0176, 0x017C, 0x8179, 0x0168, 0x816D, 0x8167, 0x0162,
        0x8123, 0x0126, 0x012C, 0x8129, 0x0138, 0x813D, 0x8137, 0x0132,
        0x0110, 0x8115, 0x811F, 0x011A, 0x810B, 0x010E, 0x0104, 0x8101,
        0x8303, 0x0306, 0x030C, 0x8309, 0x0318, 0x831D, 0x8317, 0x0312,
        0x0330, 0x8335, 0x833F, 0x033A, 0x832B, 0x032E, 0x0324, 0x8321,
        0x0360, 0x8365, 0x836F, 0x036A, 0x837B, 0x037E, 0x0374, 0x8371,
        0x8353, 0x0356, 0x035C, 0x8359, 0x0348, 0x834D, 0x8347, 0x0342,
        0x03C0, 0x83C5, 0x83CF, 0x03CA, 0x83DB, 0x03DE, 0x03D4, 0x83D1,
        0x83F3, 0x03F6, 0x03FC, 0x83F9, 0x03E8, 0x83ED, 0x83E7, 0x03E2,
        0x83A3, 0x03A6, 0x03AC, 0x83A9, 0x03B8, 0x83BD, 0x83B7, 0x03B2,
        0x0390, 0x8395, 0x839F, 0x039A, 0x838B, 0x038E, 0x0384, 0x8381,
        0x0280, 0x8285, 0x828F, 0x028A, 0x829B, 0x029E, 0x0294, 0x8291,
        0x82B3, 0x02B6, 0x02BC, 0x82B9, 0x02A8, 0x82AD, 0x82A7, 0x02A2,
        0x82E3, 0x02E6, 0x02EC, 0x82E9, 0x02F8, 0x82FD, 0x82F7, 0x02F2,
        0x02D0, 0x82D5, 0x82DF, 0x02DA, 0x82CB, 0x02CE, 0x02C4, 0x82C1,
        0x8243, 0x0246, 0x024C, 0x8249, 0x0258, 0x825D, 0x8257, 0x0252,
        0x0270, 0x8275, 0x827F, 0x027A, 0x826B, 0x026E, 0x0264, 0x8261,
        0x0220, 0x8225, 0x822F, 0x022A, 0x823B, 0x023E, 0x0234, 0x8231,
        0x8213, 0x0216, 0x021C, 0x8219, 0x0208, 0x820D, 0x8207, 0x0202
    };

    for(j = 0; j < data_blk_size; j++) {
        i = ((unsigned short)(crc_accum >> 8) ^ data_blk_ptr[j]) & 0xFF;
        crc_accum = (crc_accum << 8) ^ crc_table[i];
    }
    return crc_accum;
}

int DM2::statusError(unsigned char* buf) {
        return buf[8];
}

int DM2::length(unsigned char* buf) {
        return DM_MAKEWORD(buf[5], buf[6]) + 4;
}

int DM2::validate(unsigned char* buf) {
        if ((buf[0]!=0xFF)||(buf[1]!=0xFF)||(buf[2]!=0xFD)||(buf[3]!=0x00)) {
```

```
73              return 1;
74          }
75
76          int n = length(buf);
77          if (n < 11) {
78              return 1;
79          }
80          int err = statusError(buf);
81          if (err != 0) {
82              return err;
83          }
84
85          return 0;
86      }
87
88      // packetPrint
89      void DM2::packetPrint(int bytes, unsigned char* buf) {
90          printf("PACKET {");
91          for (int i=0; i < bytes+10; i++) {
92              printf("0x%x ", buf[i]);
93          }
94          printf("} ");
95      }
96
97      // Flush
98      void DM2::flush() {
99          while (_in.readable()) {
100             _in.getc();
101         }
102     }
103
104     // Write
105     void DM2::write(unsigned char* buf, int n) {
106         for (int i=0; i < n; i++) {
107             _out.putc(buf[i]);
108         }
109         for (int i=0; i < n; i++) {
110             _in.getc();
111         }
112     }
113
114     // Read
115     // Read reply returns payload length, 0 if error.
116     int DM2::read(int ID, unsigned char* buf, int nMax) {
```

```
117         // Broadcast doesn't reply.
118         if (ID == ID_Broadcast) {
119             return 0;
120         }
121
122         int n = 0;              // Bytes read
123         int timeout = 0; // Timeout
124         while ((timeout < 500) && (n < nMax)) {
125             if (_in.readable()) {
126                 buf[n] = _in.getc();
127                 n++;
128                 timeout = 0;
129             }
130
131             wait(_bitPeriod);
132             timeout++;
133
134             if (n > 6) {
135                 int l = length(buf);
136                 if (l < nMax) {
137                     nMax = l;
138                 }
139             }
140         }
141
142         return validate(buf);
143 }
144
145 // Send
146 // Dynamixel Communication 2.0 Protocol
147 // Header, Reserved, ID, Packet Length, Instruction, Parameter, 16bit CRC
148 int DM2::send(int ID, int bytes, unsigned char* data, unsigned char ins, unsigned char* reply) {
149         unsigned char buf[255]; // Packet
150
151         // Header
152         buf[0] = 0xFF;
153         buf[1] = 0xFF;
154         buf[2] = 0xFD;
155
156         // Reserved
157         buf[3] = 0x00;
158
159         // ID
160         buf[4] = ID;
```

```
161
162         // Packet Length
163         buf[5] = DM_LOBYTE(bytes+3);
164         buf[6] = DM_HIBYTE(bytes+3);
165
166         // Instruction
167         buf[7] = ins;
168
169         // Parameter
170         for (int i=0; i < bytes; i++) {
171               buf[8+i] = data[i];
172         }
173
174         // Checksum
175         unsigned short CRC = update_crc(0, buf, bytes+8);
176         buf[bytes+8] = DM_LOBYTE(CRC);
177         buf[bytes+9] = DM_HIBYTE(CRC);
178
179         // Transmit
180         write(buf, bytes+10);
181
182         // Read reply
183         int n = read(ID, reply);
184         if (n == 0) {
185               printf("Send: no reply");
186               return 1;
187         }
188
189         return statusError(reply); // Error code
190   };
191
192   int DM2::Test(int ID) {
193         unsigned char TxPacket[14] = {0xFF, 0xFF, 0xFD, 0x00, 0x01, 0x07, 0x00, 0x02, 0x00, 0x00, 0x00, 0x02, 0>
194         unsigned short CRC = update_crc ( 0, TxPacket , 12 ) ; // 12 = 5 + Packet Length(0x00 0x07) = 5+7
195         printf("CRC %i\n", CRC);
196         unsigned char CRC_L = DM_LOBYTE(CRC);
197         unsigned char CRC_H = DM_HIBYTE(CRC);
198         printf("CRC_L %i\n", CRC_L);
199         printf("CRC_H %i\n", CRC_H);
200
201         TxPacket[12] = CRC_L;
202         TxPacket[13] = CRC_H;
203
204         printf("TRANMITTING \n");
```

```
205         for (int i = 0; i < (14); ++i) {
206             _out.putc(TxPacket[i]);
207             _in.getc(); // Echo
208         }
209
210         wait_ms(0.5);
211
212         printf("DATA { ");
213         int timeout = 0;
214         int plen = 0;
215         while ((timeout < 100) && (plen<15)) {
216             if (_in.readable()) {
217                 printf("0x%x ", _in.getc());
218                 plen++;
219                 timeout = 0;
220             }
221
222             // wait for the bit period
223             wait(1.0/_baud);
224             timeout++;
225         }
226         printf("} \n");
227
228         return (0);
229 }
230
231 // dataPack sets data in char array and returns length.
232 int DM2::dataPack(int start, unsigned char* data, int address, int value){
233         data[start+0] = (unsigned char)DM_LOBYTE(address);
234         data[start+1] = (unsigned char)DM_HIBYTE(address);
235         data[start+2] = DM_LOBYTE(value);
236         data[start+3] = DM_HIBYTE(value);
237
238         return start + 4;
239 }
240
241 // dataPush is a generic wrapper for single value set commands.
242 int DM2::dataPush(int ID, int address, int value){
243         unsigned char data[4];
244     int bytes = 0;
245
246     bytes = dataPack(bytes, data, address, value);
247
248     return send(ID, bytes, data, INS_Write);
```

```
249 }
250
251 // Ping
252 int DM2::Ping(int ID){
253         unsigned char pong[15];
254
255         int status = send(ID, 0, NULL, INS_Ping, pong);
256         if (status != 0) {
257                 return status;
258         }
259
260         printf("PING { ");
261         for (int i = 0; i < 15; ++i){
262                 printf("PING { 0x%x", pong[i]);
263         }
264         printf("} \n");
265
266         return 0;
267 }
268
269 // SetID
270 int DM2::SetID(int ID, int newID){
271     return dataPush(ID, XL_ID, newID);
272 };
273
274 // SetBaud
275 // 0: 9600, 1:57600, 2:115200, 3:1Mbps
276 int DM2::SetBaud(int ID, int rate) {
277         if ((rate > 3) || rate < 0) {
278                 printf("Incorrect baud rate. \n");
279                 return 1;
280         }
281         return dataPush(ID, XL_BAUD_RATE, rate);
282 }
283
284 // SetLED sets motor led colours.
285 // r = 1, g = 2, y = 3, b = 4, p = 5, c = 6, w = 7, o = 0
286 int DM2::SetLED(int ID, int colour){
287         return dataPush(ID, XL_LED, colour);
288 }
289
290 // Rainbow
291 int DM2::Rainbow(int ID){
292         for (int i = 1; i < 8; ++i)
```

```
293         {
294             int status = SetLED(ID, i);
295             if (status != 0) {
296                 return status;
297             }
298             wait(1);
299         }
300         return SetLED(ID, 0);
301 }
302
303 int DM2::SetP(int ID, int value){
304         return dataPush(ID, XL_P_GAIN, value);
305 }
306 int DM2::SetI(int ID, int value){
307         return dataPush(ID, XL_I_GAIN, value);
308 }
309 int DM2::SetD(int ID, int value){
310         return dataPush(ID, XL_D_GAIN, value);
311 }
312
313 // SetGoalPosition
314 // 1024 = -150, 512 = 0 (ORIGIN), 0 = +150
315 int DM2::SetGoalPosition(int ID, int angle){
316         return dataPush(ID, XL_GOAL_POSITION_L, angle);
317 }
318
319 int DM2::SetPunch(int ID, int punch){
320         return dataPush(ID, XL_PUNCH, punch);
321 }
```

LISTING A.2: Motor driver for XL-320 (motors.cpp)

```
function delta2()
    global C S f;
    C = 0.06; % Circle Radius = Length upper arm
    S = 0.15; % Sphere Radius = Length lower arm
    f = figure;
    %zDiff = 0.8; % Base height default z-offset

    %H = [0, 0, 0]; % Sphere Head
    %B = H + [ox, oy, oz + zDiff]; % Circle Base
    HeadRadius = 0.02;
    BaseRadius = 0.037;
    ZDiffNil = 0.16;
```

```
13
14  %thetaMax =
15  swingMax = 45/180*pi;
16
17  v1 = [1.0, 0.0, 0.0];
18  v2 = [-0.5, sqrt(3)/2, 0.0];
19  v3 = [-0.5, -sqrt(3)/2, 0.0];
20
21  head1 = v1*HeadRadius;
22  head2 = v2*HeadRadius;
23  head3 = v3*HeadRadius;
24
25  base1 = v1*BaseRadius;
26  base2 = v2*BaseRadius;
27  base3 = v3*BaseRadius;
28
29  plane1 = [0.0, 1.0, 0.0];
30  plane2 = [sqrt(3)/2, 0.5, 0.0];
31  plane3 = [-sqrt(3)/2, 0.5, 0.0];
32
33  arm45 = S*cos(swingMax);
34
35  %% Solve for 3 points
36  array = -0.16:0.16/40:0.16;
37
38  for oz = -0.16:0.16/40:0.04
39      for oy = array
40          MAX = false;
41          MIN = false;
42          for ox = array
43              offset = [-ox, -oy, -oz+ZDiffNil];
44              b1 = base1 + offset;
45              p1 = deltaSolver(head1, b1, plane1); %p1
46              if (p1(3) > b1(3)) || (distToPlane(p1, head1, plane1) > arm45) || not(isreal(p1))
47                  continue
48              end
49
50              b2 = base2 + offset;
51              p2 = deltaSolver(head2, b2, plane2); %p2
52              if (p2(3) > b2(3)) || (distToPlane(p2, head2, plane2) > arm45) || not(isreal(p1))
53                  continue
54              end
55
56              b3 = base3 + offset;
```

```
            p3 = deltaSolver(head3, b3, plane3); %p3
            if (p3(3) > b3(3)) || (distToPlane(p3, head3, plane3) > arm45) || not(isreal(p1))
                continue
            end

            if MIN
                max = [ox, oy, oz];
                MAX = true;
            else
                plotter([ox, oy, oz], f);
                MIN = true;
            end
        end
        if MAX
            plotter(max, f);
        end
    end
  end
end
function d = distToPlane(p, h, plane)
    hp = p - h;
    d = dot(plane, hp)/norm(plane);
end
function pnt = deltaSolver(h, b, p)
    global C S f;
    cs = h - b;
    sm = dot(p,cs)*p; % Sphere to projection point m

    cm = cs - sm; % Circle to projection point m

    s = sqrt(S^2-norm(sm)^2); % Radius sphere projection

    i = cm/norm(cm);
    j = cross(p,i);


    a = norm(cm);

    %x2+y2=Rj2 (a-x)2+y2=r2
    x = (a^2-s^2+C^2)/(2*a);

    y = sqrt(C^2-x^2);

    p1 = b+i*x+j*y;
```

```
    p2 = b+i*x-j*y;
    if norm(p1) > norm(p2);
        %plotter([b;p1;h],f);
        pnt = p1;
    else
        %plotter([b;p2;h],f);
        pnt = p2;
    end;
end
%% Plot
function plotter(v,f)
    figure(f)
    hold on
    view(3)
    scatter3(v(:,1),v(:,2),v(:,3), [], v(:,3));

end

%% Circle
function originCircle(r, p)
    global f
    % Original points, original plane
    t = linspace(0,2*pi);
    x = r*cos(t) + p(1);
    y = r*sin(t) + p(2);

    z = 0*t+p(3);
    pnts = [x;y;z];

    figure(f)
    hold on
    view(3)
    plot3(pnts(1,:),pnts(2,:),pnts(3,:))
    axis equal
end
```

LISTING A.3: Delta arm build plate

```
1  package delta;
2
3  message Message {
4          enum Type { ERROR = 1; START = 2; STOP = 3; PING = 4; POINT = 5; }
5
6          // Type Identifier

7          required Type type = 1;
8
9          // Supported types
10         optional Command error = 2;
11         optional Command start = 3;
12         optional Command stop = 4;
13         optional Command ping = 5;
14         optional Point point = 6;
15 }
16
17 message Command {
18         optional string id = 1;
19 }
20
21 message Point {
22         required double x = 1;
23         required double y = 2;
24         required double z = 3;
25 }
```

LISTING A.4: Google Protocol Buffer basic Message structure

Quadcopter flying platforms suffer from small instabilities and drift in position, similar to all flying vehicles. Maintaining a fixed reference point to the ground is challenging as forces acting in disturbance, such as wind and ground effects, cannot be absorbed through a physical medium and instead must be accounted through thrust vectoring, ie. using flight control to hover precisely in place. However, quadcopters control is inherently reactive to disturbance, and state estimation is only able to predict a position within an error margin. It therefore can only maintain holding position within a certain error margin required to perform construction designs. Using the best platforms available a hover precision of about 7cm can be reached while hovering close to ground (about 20cm from ground). This precision is not sufficient for aerial 3d printing, i.e. using a flying vehicle to deposit material precisely in place. Possibilities to stabilise the platforms include (i) Gimbal systems as used for cameras that currently use to correct for motion in flight and remove vibrations by controlling the rotation, (ii) x-y stages that can be attached to the platform, (iii) a delta arm that is integrated on the platform and (iv) a soft trunk that is attached to the platform.

This work outlines the use of such manipulation devices that are integrated on an aerial vehicle with the purpose of using it to precisely deposit material for aerial 3d printing to precisely manipulate objects or use drills and other mechanical tools to fabricate material while the platform is in flight. To achieve the required precision a decoupling of the movement of the aerial platform and the device is required and possibly a separate electronics and sensor pack needs to be integrated to separately stabilise the platform and the manipulator.

Advantages of using an aerial vehicle to manipulate, 3d print and machine objects:

- Flexibility: Ability to operate dynamically in space, especially in difficult to access areas such as high altitudes, pipelines, buildings, bridges, roofs etc.

- Efficiency and collaborative construction and servicing: Pre-programmed robots that are able to work in cooperation will speed up the construction process.

- Precision: Robots can precisely follow trajectories and complicated tasks using visual feedback or external tracking.
- Consistency: Provide the ability to carry out construction with the same standard
- Safety: Eliminates dangers for human workers, high altitude and dangerous terrain.

A two-bodied system is formed from the decoupling achieved between the quadcopter and device. The devices absolute position is now removed from marginal errors produced from the quadcopter. There are different systems to decouple the device in the form of an XYZ stage, XY gimbal with Z radial depth, 3-axis articulated robotic arms and Delta arm. As a first step we decided to prototype a delta arm, consisting of a parallel robot manipulator with three motors connected to arm linkages. The parallel second arm stage enforces orientation is maintained. Motors are maintained within the base of the robot allowing the arms weight to be minimal, thus greater movement acceleration over a conventional robotic arm. This is important for maintaining a stable reference point and reducing the torque forces passed to the drone. We also developed modelling and control principles that allow the precise control of these arms.

Figure 14: Modelling of Delta arm robots.

Stabilisation techniques exist on small quadcopters and other flying vehicles, however there is a fundamental difference in the type of stabilisation achieved. Existing solutions aim to reduce the vibration and planar rotations. This allows devices, such as cameras and other recording equipment, to maintain a steady "look at" position. Absolute position is not controlled. The delta arm robot would provide absolute position stabilisation. This allows devices to act from a defined reference position enabling interaction on other reference points.

An add-on delta arm proof-of-concept prototype is currently under development. A prototype device will be retrofitted to a drone allowing stabilisation in flight. Further development would incorporate additional features such as a foam dispensing mechanism that will allow for the creation of 3D structures. A foam-dispensing mechanism has already been developed and it would be used in conjunction to provide a set of useful tools showing the full potential of the device.

This project aims to create a fully working prototype that will include the following systems: Robotic arm linkages, 3d printing module, motor actuator, control system and a programmable interface providing wireless control.

Commercial applications of robotic stabilisation with flying robots include:

- Humanitarian
    - Search and rescue, accurate control allows for manoeuvre of objects, provisioning of aid.
    - Emergency rescue shelter construction.
- Aeronautical Engineering
    - Mid flight repairs
- Civil Engineering
    - Robotic construction

- Building repairs reduce response time allowing for emergency repairs and difficult inaccessible areas compared to conventional ground based techniques.

- Military

5

- Remote autonomous bomb removal/disposal.
- Emergency structure construction.

Claims

1. An aerial device capable of controlled flight, the aerial device comprising:

a first body comprising a lift generator for providing lift to the aerial device;

a second body comprising a substance dispenser for controllably dispensing a curable substance for additive manufacture;

an articulated coupling assembly coupling the first body to the second body;

an actuator operable to articulate the articulated coupling assembly so as to move the second body, and thereby the substance dispenser, relative to the first body; and

a controller operable to control: the lift generator, the substance dispenser, and the actuator.

2. The aerial device of claim 1, wherein at least a portion of the actuator is fixedly coupled to the first body.

3. The aerial device of claim 2, wherein the articulated coupling assembly comprises three arms arranged, together with the first body and second body, to form a delta robot.

4. The aerial device of any preceding claim, wherein the controller is further operable to receive position-related information and to control the actuator based on the received information.

5. The aerial device of claim 4, wherein the position-related information comprises image information from one or more cameras.

6. The aerial device of claim 5, further comprising the one or more cameras.

7. The aerial device of any preceding claim, further comprising an accelerometer arranged to produce acceleration-related information and provide that information to the controller, and further wherein the controller is operable to control the actuator based on the acceleration-related information.

8. The aerial device of any preceding claim, wherein the controller is further operable to:

determine a position of the first body or the second body relative to a target, and

based on the determined position, control the actuator in order to move the second body towards the target.

9. The aerial device of claim 8, further comprising an energy store, and wherein the target is a charger operable to provide power to the energy store.

10. The aerial device of any preceding claim, wherein the controller is further operable to detect an object and to control the actuator in order to catch the object.

11. The aerial device of any preceding claim, wherein the controller is further operable, when the second body is resting on a surface, to control the actuator in order to jump the aerial device relative to the surface.

12. A method of operating the aerial device of any of claims 1 to 11.

13. A computer-readable medium carrying computer-readable instructions arranged, upon execution by a processor, to cause the processor to carry out the method of claim 12.

14. A computer-readable medium having data stored thereon representative of the aerial device according to any of claims 1 to 11, the data being such that it can be relayed to an additive manufacturing device to enable the additive manufacturing device to fabricate the aerial device based on the data.

15. A method, system, or apparatus substantially as described herein and with reference to the appended figures.

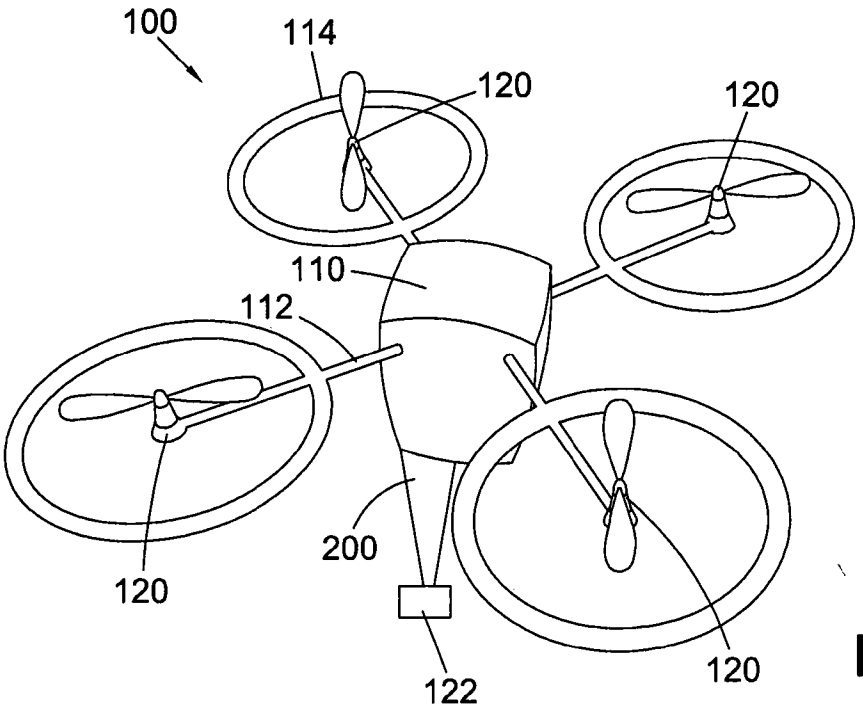

100
114
120
120
110
112
200
120
120
122

Fig. 1

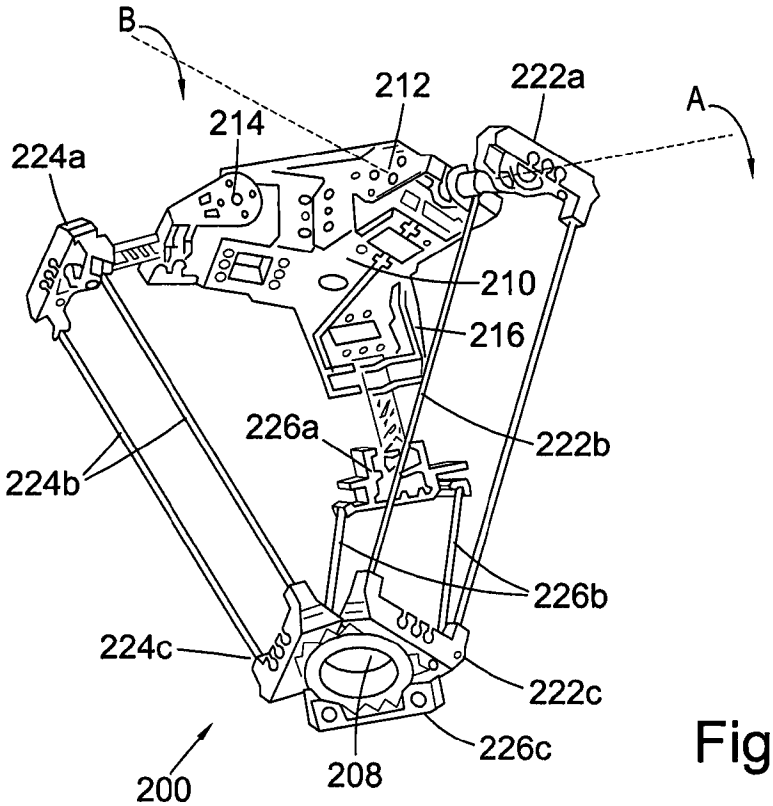

B
212
222a
A
214
224a
210
216
226a
222b
224b
226b
224c
222c
226c
208
200

Fig. 2

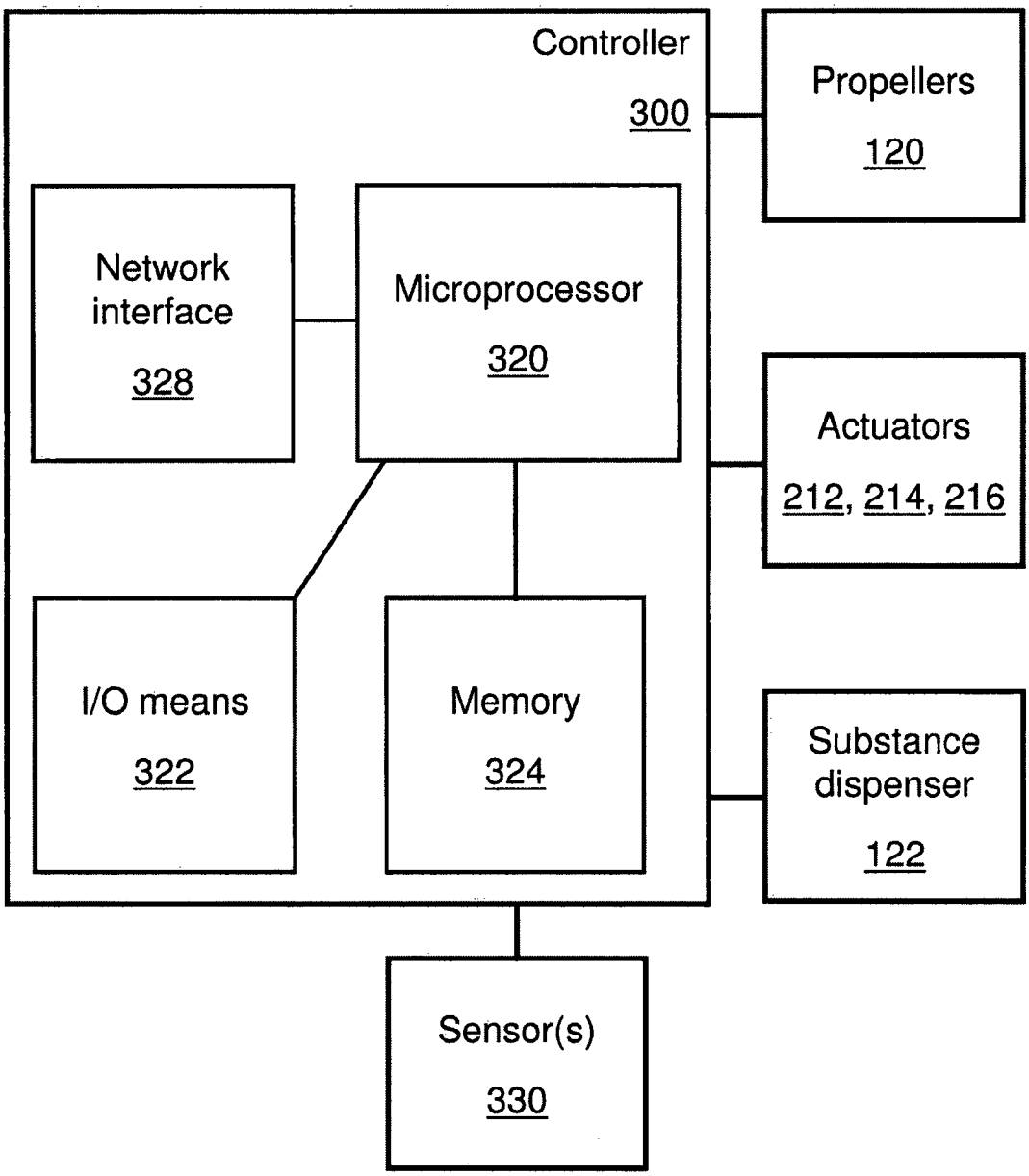
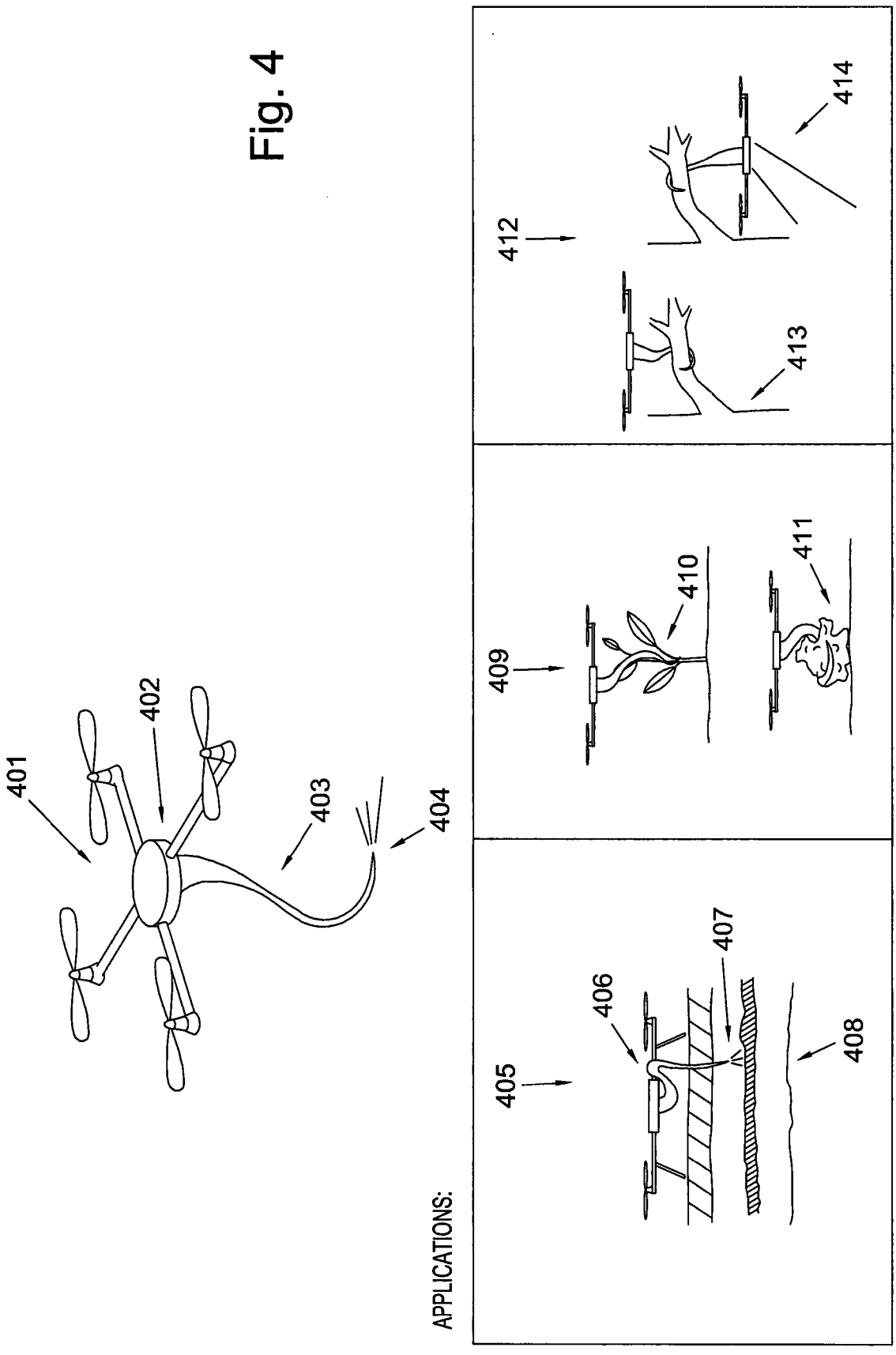

Controller
300
Propellers
120
Network interface
328
Microprocessor
320
Actuators
212, 214, 216
I/O means
322
Memory
324
Substance dispenser
122
Sensor(s)
330

Fig. 3

Fig. 4


401
402
403
404
APPLICATIONS:
405
406
407
408
409
410
411
412
413
414

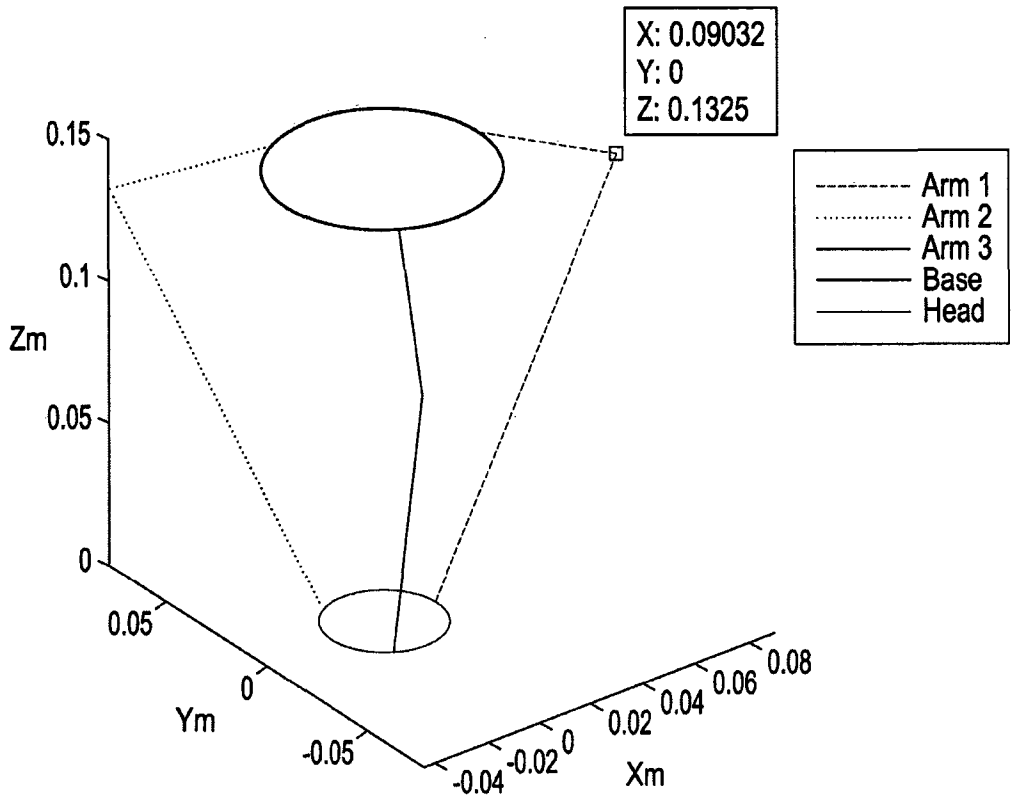

X: 0.09032
Y: 0
Z: 0.1325
Arm 1
Arm 2
Arm 3
Base
Head
0.15
0.1
0.05
0
Zm
0.05
0
-0.05
Ym
-0.04
-0.02
0
0.02
0.04
0.06
0.08
Xm

Fig. 5

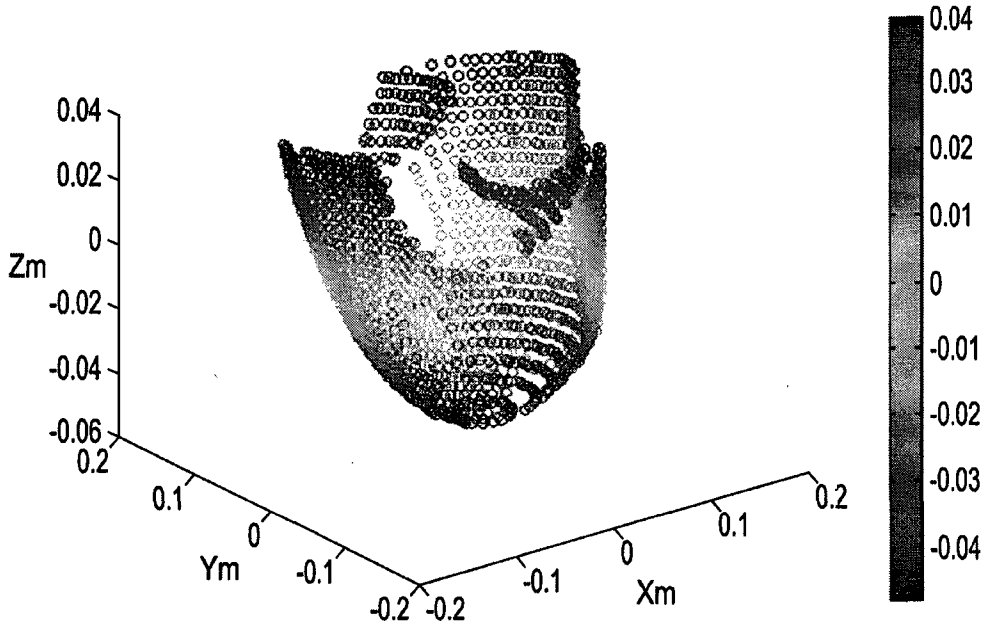

0.04
0.02
0
-0.02
-0.04
-0.06
Zm
0.2
0.1
0
-0.1
-0.2
Ym
-0.2
-0.1
0
0.1
0.2
Xm
0.04
0.03
0.02
0.01
0
-0.01
-0.02
-0.03
-0.04

Fig. 6

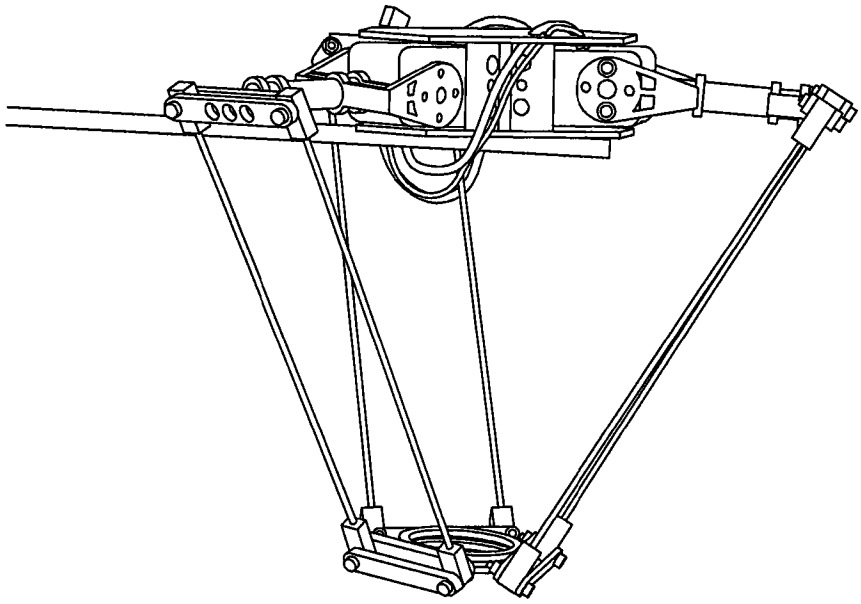


Fig. 7

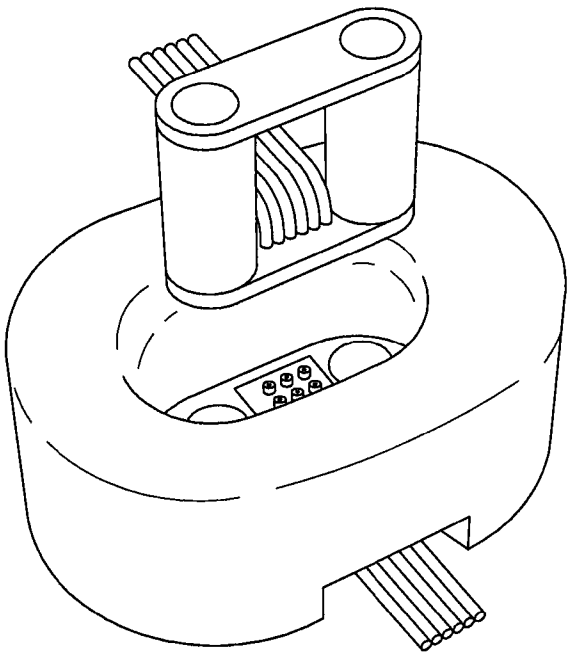


Fig. 8

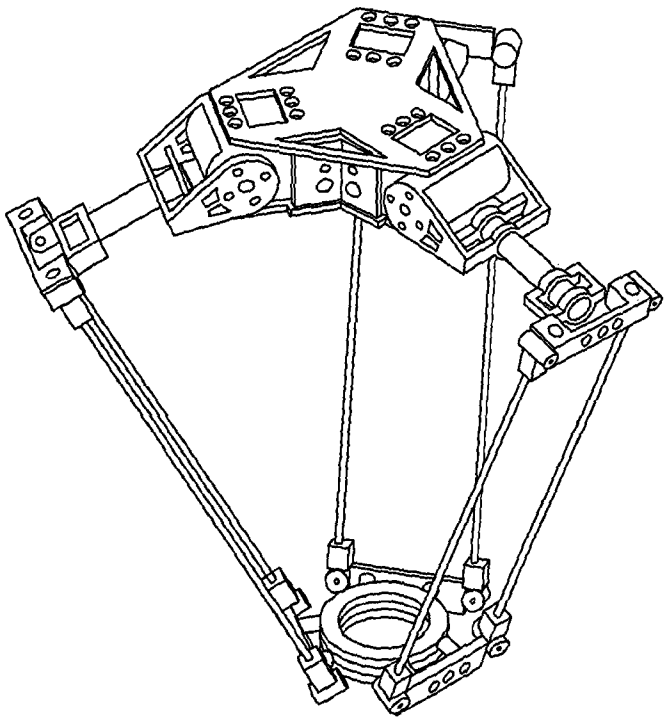


Fig. 9

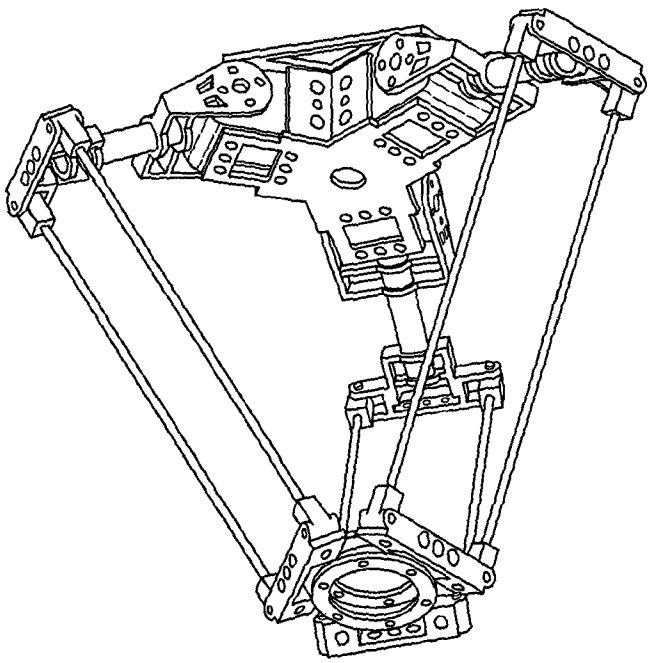


Fig. 10

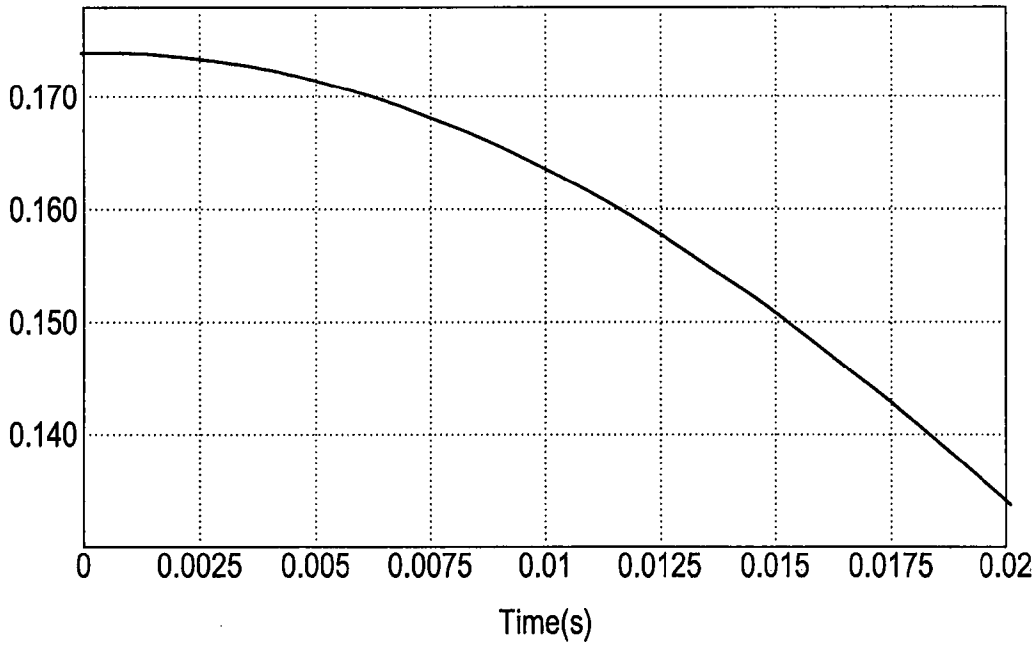

0.170
0.160
0.150
0.140
0
0.0025
0.005
0.0075
0.01
0.0125
0.015
0.0175
0.02
Time(s)

Fig. 11

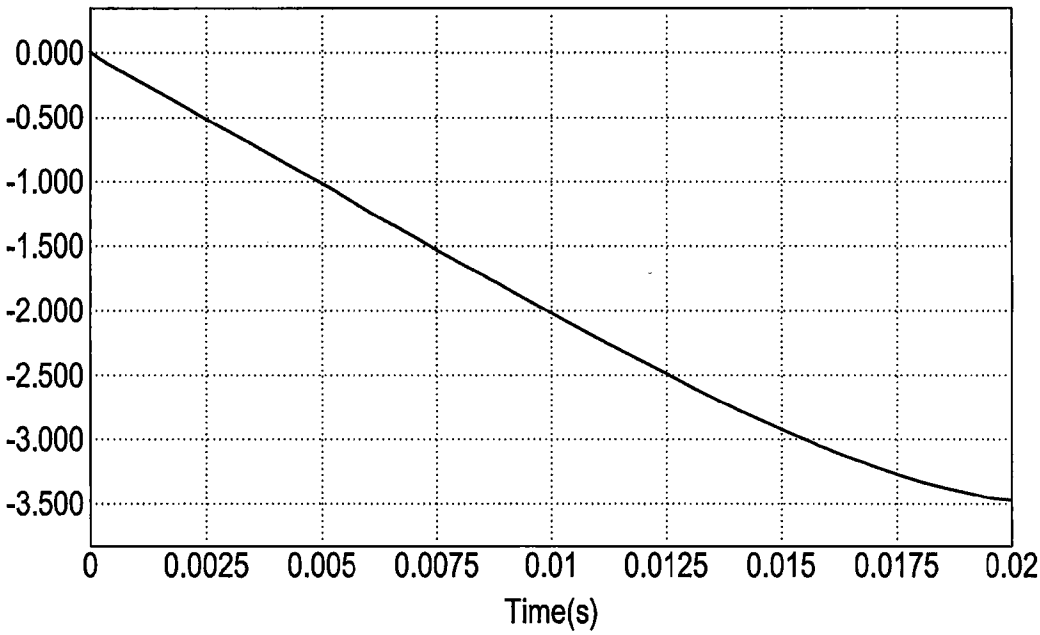

0.000
-0.500
-1.000
-1.500
-2.000
-2.500
-3.000
-3.500
0
0.0025
0.005
0.0075
0.01
0.0125
0.015
0.0175
0.02
Time(s)

Fig. 12

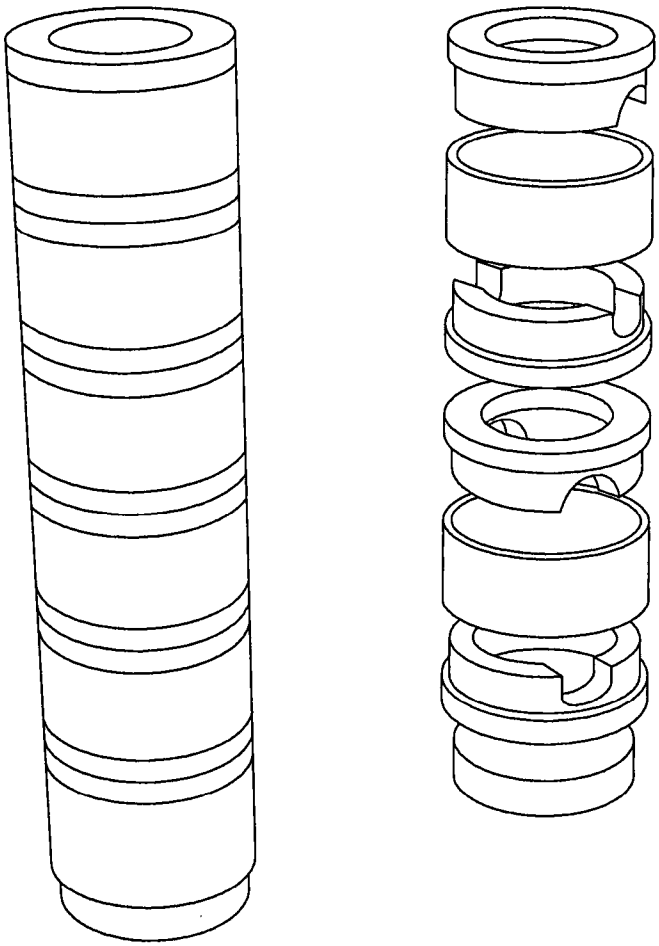


Fig. 13

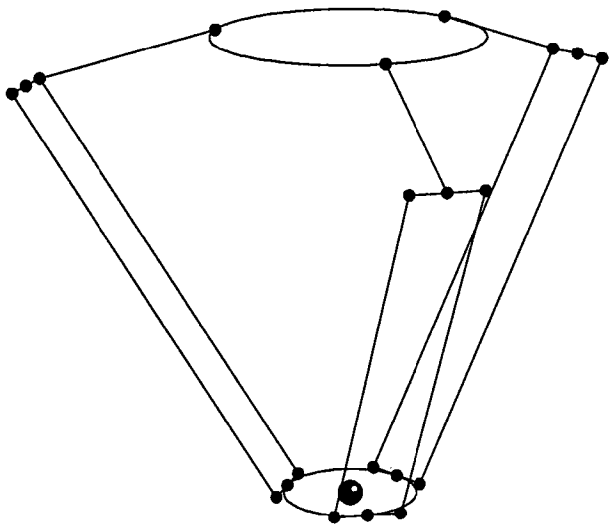


Fig. 14

INTERNATIONAL SEARCH REPORT

International application No
PCT/GB2016/051427

A. CLASSIFICATION OF SUBJECT MATTER
INV. B64C39/02
ADD.

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
B64C B29C A63H B33Y B28B B25J

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
EPO-Internal, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	HUNT GRAHAM ET AL: "3D printing with flying robots", 2014 IEEE INTERNATIONAL CONFERENCE ON ROBOTICS AND AUTOMATION (ICRA), IEEE, 31 May 2014 (2014-05-31), pages 4493-4499, XP032650080, DOI: 10.1109/ICRA.2014.6907515 [retrieved on 2014-09-22]	1,2,4-14
Y	page 4493 - page 4498	3
	----- -/--	

[X] Further documents are listed in the continuation of Box C.

[X] See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
12 August 2016	23/08/2016

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Cetiner-Fresneda, B

4

Form PCT/ISA/210 (second sheet) (April 2005)

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	Rowan Hooper: "Flying 3D printer could seal off nuclear waste ¦ New Scientist", , 7 May 2014 (2014-05-07), pages 1-2, XP055286291, Retrieved from the Internet: URL:https://www.newscientist.com/article/mg22229683-900-flying-3d-printer-could-seal-off-nuclear-waste/ [retrieved on 2016-07-06]	1,2,4-8, 12
A	Video; the whole document	3
Y	Deltabot: "3ders.org - Rostock: an amazing delta robot 3D printer prototype ¦ 3D Printing news", , 14 July 2012 (2012-07-14), pages 1-5, XP055286392, Retrieved from the Internet: URL:http://www.3ders.org/articles/20120714-rostock-an-amazing-delta-robot-3d-printer-prototype.html [retrieved on 2016-07-06]	3
A	the whole document	14
A	US 2014/022051 A1 (LEVIEN ROYCE A [US] ET AL) 23 January 2014 (2014-01-23) paragraph [0204]; figure 31	9
A	EP 2 003 057 A2 (HONEYWELL INT INC [US]) 17 December 2008 (2008-12-17) figure 6	10
A	JP H07 24751 A (TOSHIBA CORP; TOSHIBA ENGINEERING CO) 27 January 1995 (1995-01-27) page 5, column 2, lines 31-34	11
A	WO 2014/200604 A2 (GILMORE ASHLEY A [US]; DEWEY DAVID L [US]) 18 December 2014 (2014-12-18) paragraph [0134]	13

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/GB2016/051427

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2014022051 A1	23-01-2014	US 2014022051 A1	23-01-2014
		US 2014022055 A1	23-01-2014
		US 2014025229 A1	23-01-2014
		US 2014025230 A1	23-01-2014
		US 2014067159 A1	06-03-2014
		US 2014067160 A1	06-03-2014
		US 2014067167 A1	06-03-2014
EP 2003057 A2	17-12-2008	EP 2003057 A2	17-12-2008
		US 2009050750 A1	26-02-2009
JP H0724751 A	27-01-1995	NONE	
WO 2014200604 A2	18-12-2014	EP 2972462 A2	20-01-2016
		US 2015205301 A1	23-07-2015
		WO 2014200604 A2	18-12-2014