Autonomous Quadrotor Delivery System Modeling and Simulation

A.M. El-Edkawy and M.A. El-Dosuky
Mansoura University, Faculty of Computers and Information, Mansoura, 35516, Dakahlia, Egypt
amr.eledkawy@mans.edu.eg, mouh_sal_010@mans.edu.eg

Abstract: *A quadrotor is a rotary-wing Unmanned Aerial Vehicle (UAV) that has four rotors. The quadrotor control and autonomy is a challenging task as the quadrotor system is considered underactuated system. The aim of this paper is to present the specification and the implementation of an autonomous quadrotor delivery system that fuses data from camera and IMU (Inertial Measurement Unit) to allow for high precision visual-inertial navigation. In this paper, an autonomous quadrotor delivery system is proposed. The specification of the system is proposed. The implementation of the system introduced SolidWorks design for the system, mathematical modelling for the pickup and delivery problem (PDP). Simulation results using MATLB have been shown for the implementation of the model, control system and the simultaneous localization and mapping (SLAM) using Extended Kalman Filter algorithm (EKF-SLAM) used for making the system autonomous. Experimental results for the use of canny edge detection algorithm have been shown.*

Keywords: *Quadrotor; UAV; PDP; PID controller, SLAM; EKF-SLAM; Canny Edge Detection*

1 Introduction

A quadrotor is a rotary-wing Unmanned Aerial Vehicle (UAV) [1] that has four rotors. To enable a quadrotor to navigate autonomously in outdoor environments is highly challenging. A quadrotor is difficult to control, since it is considered as an underactuated system [2, 3] for it has six Degrees of Freedom (DOF) but only four actuators.
Fusing inertial measurements and features in images captured from camera is known as a vision-aided inertial navigation [4, 5].

The aim of this paper is to present the specification and the implementation of an autonomous quadrotor delivery system that fuses data from camera and IMU (Inertial Measurement Unit) to allow for high precision visual-inertial navigation.

In this paper, an autonomous quadrotor delivery system is proposed. The specification of the system is proposed. The implementation of the system introduced SolidWorks design for the system, mathematical modelling for the pickup and delivery problem (PDP) [6, 7]. Simulation results using MATLAB have been shown for the implementation of the model, control system and simultaneous localization and mapping (SLAM) using Extended Kalman Filter algorithm (EKF-SLAM) [8] used for making the system autonomous. Experimental results for the use of canny edge detection algorithm [9, 10] have been shown.

Later in the paper, second section explains the background and the previous work about the design requirements for a delivery quadcopter, the modelling and the control of it, how to make use of computer vision algorithm and how to make the system autonomous. Third section illustrates the proposed system by providing the specification of it and drawing a general framework explaining it then introduce the SolidWorks model and the mathematical modeling of PDP. Fourth section shows the simulation and results developed using MATLAB. Fifth section concludes the paper and introduces the future directions based on the proposed system.

2 Previous Work

Several big firms are eager to utilize flying robots in delivery.
Amazon: Amazon Prime Air is a conceptual drone-based delivery system currently in development by Amazon (https://www.youtube.com/watch?v=Le46ERPMlWU) .
Domino's: Domino's has tested out a prototype for a flying drone that can deliver pizzas. The device carried two large pepperoni pizzas and conducted a flight (https://www.youtube.com/watch?v=-CYT4PFV_Hs).
Francesco's Pizzeria: Francesco's Pizzeria provides a way for delivering pizza using flying robots (https://www.youtube.com/watch?v=0if2PM6OBrI.

2.1 Design requirements

In a perfect world, the quadcopter would be equipped for independent flight through corridors, rooms and stairwells in any building. In the building, the quadcopter ought to be fit for transmitting data about the environment remotely to a remote first reaction group. The following is the key outline necessities for a delivery quadrotor UAV [11]:

1. Hover at 40-50% of maximum throttle (1.5:1-2:1 thrust to weight ratio).
2. Life of battery is no less than 10 minutes.
3. Autonomous take off, hover, traversal and landing.
4. Sensors for estimating relative separations from remote objects, height and odometry.
5. Maximum width under 30" (normal width of a gate).
6. Wireless availability.
7. Insignificant flight elevation of nearly 5-7ft, maximum height of no less than10 feet.
8. Whole mass under 1.5kg, for a possible load of 0.5kg-1kg
9. Protection on the propellers with the end goal that the quadcopter can survive 10 foot falls.

There are few other options to creating and developing craft that worth considering, however everyone has its own downsides. Most eminently, commercial UAVs tend to have a few fundamental classes of weaknesses. They have a tendency to be intended to convey no extra payload other than themselves (for instance the customer level Draganflyer, Parrot AR.Drone, Silverlit X-UFO, and so forth.), while those that can convey extra are restrictively costly (Microdrone MD4-200, modern Draganflyer, AscTec), or they are too huge to work in our objective usee case. Open implementations, for example, the Aeroquad and MikroKopter can, through broad modification in the design, be near to our coveted parameters but would not spare the designer significant effort and would limit the long term value, adaptability and lifetime of the platform [11].

2.1 Quadcopter modelling and control

The Quadrotor primary thought was to have a helicopter that can deal with bigger limit and can move in zones that were hard to reach. It comprises four propellers (right, left, front and back,). Varying the speeds of the engines is utilized to control the quadrocopter in each of the three axes. As shown in Figure 1, the left and right engines rotate counterclockwise while the front and back propellers rotate clockwise. This adjust the aggregate framework torque and wipe out the aerodynamics and gyroscopic torques in stationary flights [12].

Yaw: Changing the rpms of the sets of counter rotating engines, the quadrocopter will yaw because of the created moment.

Roll: Increasing and decreasing the rpms of the left and right engines make the quadrocopter rolled forward or backward.

Pitch: Same as roll but using front and back engines instead of left and right.

Figure 1: Roll, Pitch and Yaw

There are many papers that discussed the modelling of the Quadrotor [13-17]. In [17] the concluded final result for the body frame rotational equations of motion for the quadrotor is the following:

$$\dot{\omega} = \begin{bmatrix} \tau\varphi {I_{xx}}^{-1} \\ \tau\theta {I_{yy}}^{-1} \\ \tau\psi {I_{zz}}^{-1} \end{bmatrix} - \begin{bmatrix} \frac{I_{yy}-I_{zz}}{I_{xx}}\omega_y\omega_z \\ \frac{I_{zz}-I_{xx}}{I_{yy}}\omega_x\omega_z \\ \frac{I_{xx}-I_{yy}}{I_{zz}}\omega_x\omega_y \end{bmatrix} \quad (1)$$

where τ is the motor torque, I is the input current, $\varphi, \theta \; and \; \psi$ are the roll, pitch, yaw in the body frame respectively, $x, y \; and \; z$ are determining the position of the quadrotor, ω is the angular velocity vector in the body frame.

In order to do a real world simulation we must do a quadrotor control, clarifies Proportional-Integral-Derivative (PID) controller controls the position, attitude and altitude of the quadrotor. The following figure clarifies PID controller [14].

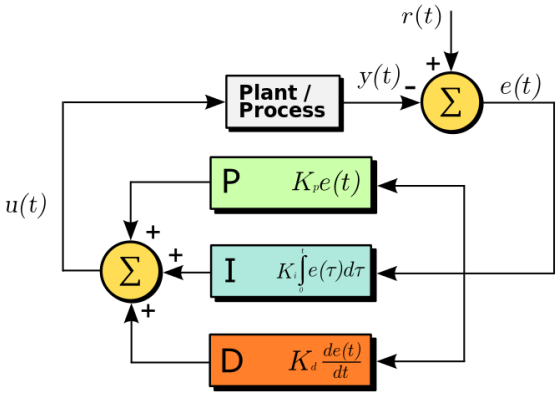

Figure 2: PID Controller

2.2 Computer Vision

Canny edge detection algorithm is a popular multi step edge detection algorithm starting with input image $f(u, v)$ of size $u \times v$ pixels. It has the following steps [9, 10].

1. Apply Gaussian filter G_a to smooth the image $f(u,v)$ in order to remove the noise

$$gr(u,v) = G_a(u,v) * f(u,v) \quad (2)$$

where

$$G_a = \frac{1}{\sqrt{2x\sigma^2}} \exp\left(-\frac{u^2+v^2}{2\sigma^3}\right) \quad (3)$$

2. Find the intensity gradients $M(u,v)$ of the smoothed image $gr(u,v)$

$$M(u,v) = \sqrt{gr_u^2(u,v) + gr_v^2(u,v)} \quad (4)$$

and the gradient direction

$$\theta(u,v) = \tan^{-1}\{gr_v(u,v)/gr_u(u,v)\} \quad (5)$$

3. Threshold

$$M_T(u,v) = \begin{cases} M(u,v) \; if \; M(u,v) > T \\ 0 \qquad otherwise \end{cases} \quad (6)$$

4. Apply non-maximum suppression to eliminate fake response to edge detection
5. Apply double threshold to determine possible edges
6. Apply hysteresis: Finalize edge detection by eliminating weak edges.

2.3 Autonomous Robot

The block diagram of control unit of most autonomous robots is shown in figure 3.

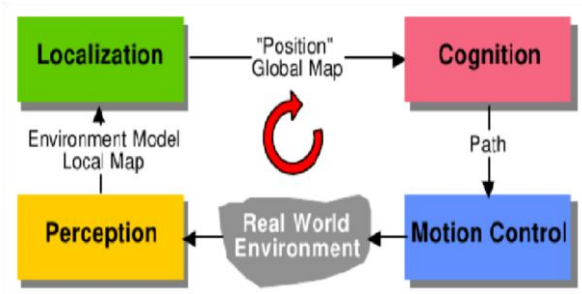

Figure 3: Control unit of autonomous robots [18]

To be autonomous, the quadrotor has to solve the simultaneous localization and mapping (SLAM) problem [8]. So it needs to decide its location inside a varying map that it constructs from the environment established on data synthesis [19] from numerous noisy data sources [1]. Implementations exploit sonar [20] or visual navigation [21]. The regular used filter is Kalman Filter [22].

RaoBlackwellized particle filter (FastSLAM) and Extended Kalman filters (EKF-SLAM) are clarified in [8]. In stochastic mapping [23], data association is is basically tended to utilizing a traditional procedure in following issues known as the nearest neighbor (NN) [24]. The development of Joint Compatibility Branch-and-Bound data association (JCBB) [25] is is supported by the way that the basic NN algorithm for data association is exceptionally delicate to the sensor mistake, expanding the likelihood of coordinating unrelated map features [25].

3 Proposed System

3.1 Specification and Setup

PEAS (Performance measure, Environment, Actuators, and Sensors) [26] is used to determine the specifications of an agent. In the proposed system, the agent is the quadrotor. Table 1 shows the PEAS specifications of the quadcopter.

Table 1: PEAS specifications of the quadcopter

Performance Measure	Reaching destination quickly and correctly
Future Performance Measure	Collaboration among quadrotors as a swarm
Environment	Roads, Buildings and objects
Actuators	4 rotors
Sensors	Camera and Inertia Measurement Unit (IMU)

ODESA (Observable, Deterministic, Episodic, Static, and Agents) [26] is used to describe the environment characteristics. Table 2 specifies the characteristics of the environment of the quadcopter.

Table 2: ODESA specifications of the quadcopter environment

Observable	Partial Observable
Deterministic	Stochastic
Episodic	Sequential
Static	Dynamic
Agents	Multi-agent

Figure 2 shows the block diagram of the system which shows an overall view of the system

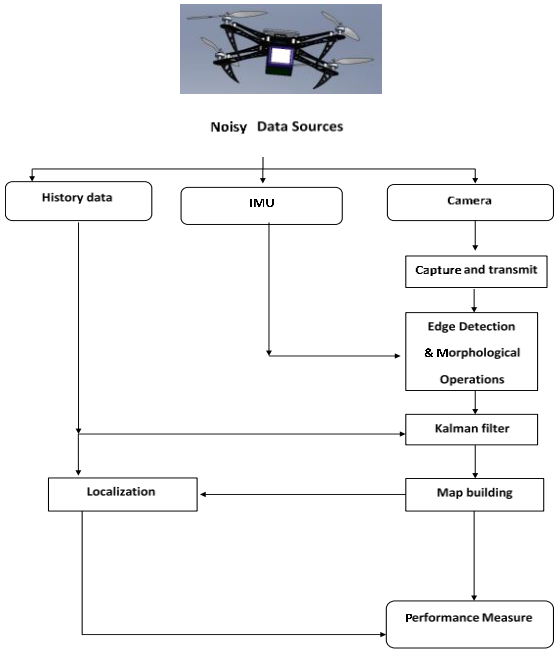

Figure 4: Proposed System Block Diagram

3.2 Modelling and Design

Figure 3 demonstrates our 3D model of the quadrotor in SolidWorks. This model have the following components.

Table 3: Components of the 3D model

2 Bodies
Arduino-UNO
Battery 4Ah 5s Lipo Nano-Tech Jr.
Spectrum receiver BR 60000 Kevin Barker
4 Propellers
4 Motors 1300Kv
4 Above Paws
4 Propeller Screws
4 Down Paws
4 Paws
4 Block Paws
12 Separators
Camera
IMU

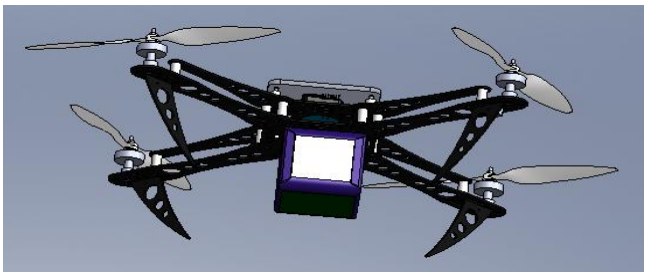

Figure 5: SolidWorks Model

The model of the quadrotor is enhanced using the quadrotor route finding problem that is a variant of a pickup-and-delivery problem (PDP). It is expressed in [6] as a linear program that have many constraints in the delivery system.

3.3 PDP

The quadrotor route finding problem is a variant of a pickup-and-delivery problem (PDP). It is expressed here as a linear program. We will make use of the following sets:

- V is the set of all ground vehicles (base stations for the quadcopters).
- H is the set of all quadcopters.
- S is the set of all items available in the request system.
- R is the set of requests that must be served.
- $N = H \cup S \cup V$ is the collection of all important locations
- $E = N^2$ is the set of directed edges in the graph on N

  For $e \in E$, we will denote the distance between the endpoints of the edge by $|e|$. Sometimes, we will refer to an edge by its ordered pair. The variables of the linear program are:

- $\{x_e^h\}_{e\in E, h\in H}$ Binary variables with value 1 if quadcopter h traverses edge e and 0 otherwise
- $\{q_e^s\}_{e\in E, s\in S}$ Quantity of item s carried along edge e (integer)
- $\{z_e^h\}_{e\in E}^{h\in H}$ Quantity representing the state of the battery of quadcopter h after traversing edge e
- $\{t_e^h\}_{e\in E, h\in H}$ Abstract quantity roughly quantifying the number of legs remaining in the trip for quadcopter h after traversing edge e.
- T Time required for all quadcopters to complete their routes

The objective function is then easily expressed as, $f = T$

Constraints

There are a large number of constraints, so we will present the bound constraints:

- $\forall h \in H, e \in E, \quad 0 \le x_e^h \le 1$
- $\forall s \in S, e \in E, \quad 0 \le q_e^s$
- $\forall h \in H, \forall e \in E \quad 0 \le z_e^h$
- $\forall h \in H, e \in E, \quad 0 \le t_e^h$
- $\forall h \in H, v \in N, \quad x_{(v,v)} \le 0$ (Eliminate self-loops)
- $\forall h \in H, v' \in N, \quad x_{(v',v)}^h \le 0,$ where quadcopter h starts at v. (No one can fly to the starting location of a quadcopter)
- $\forall h \in H, v' \in N, \quad z_{(v,v')} \le c - \frac{|(v,v')|}{maxRange},$ where quadcopter h starts at v with initial charge, c. (Remaining battery after the first leg is bounded by the initial charge less the battery consumed on the first leg)
- $\forall v \in V, v' \in N, \quad z_{(v,v')}^h \le 1 - \frac{|(v,v')|}{maxRange(h)}$ (Remaining battery after departing a station is bounded by a full charge, less the battery consumed in departing the station)
- $\forall h, h' \in H, v' \in N, x_{v,v'}^{h'} \le 0,$ where quadcopter h starts at v and $h \neq h'$. (No quadcopter may depart from another quadcopter's starting location)

The remaining equality and inequality constraints are broken up into a number of types.

- Type 0 constraints. Each quadcopter can only leave its starting location in one direction.

  $$\forall h \in H, \quad \sum_{e=(v,v')\in E} x_e^h \le 1,$$

  where quadcopter h starts at v.

- Type 1 constraints. The quantity of each item carried away from each quadcopter's starting location is equal to the quantity of that item that particular quadcopter is currently carrying.

  $\forall h \in H, s \in S, v' \in N,$

  $q_{(v,v')}^s = x_{(v,v')}^h q_{init}(h),$ where quadcopter h starts at v

- Type 2 constraints. No route is flown twice.

$$\forall e \in E, \qquad \sum_{h \in H} x_e^h \leq 1$$

- Type 3 constraints. Every request is serviced.

$$\forall r \in R, s \in S, \qquad \sum_{e=(v',v) \in E} q_e^s - \sum_{e=(v,v') \in E} q_e^s = \#\text{ of items of type } s \text{ in request } r$$

Here we use the convention that a delivery is a positive number of items and a pickup is negative. Also, v corresponds to the location of request r.

- Type 4 constraints. Every request is visited once.

$$\forall r \in R, \quad \sum_{h \in H} \sum_{e=(v',v) \in E} x_e^h = \sum_{h \in H} \sum_{e=(v,v') \in E} x_e^h = 1$$

, where v is the location of request r

- Type 5 constraints. Cargos must respect the carrying capacity of the quadcopters.

$$\forall e \in E, \qquad \sum_{s \in S} q_e^s weight(s) - \sum_{h \in H} x_e^h capacity(h) \leq 0$$

- Type 6 constraints. A quadcopter moves to a request if and only if it moves away.

$$\forall r \in R, h \in H, \quad \sum_{e=(v',v) \in E} x_e^h - \sum_{e=(v,v') \in E} x_e^h = 0$$

where v is the location of request r

- Type 7 constraints. A quadcopter arrives at each station at least as many times as it leaves.

$$\forall h \in H, v \in V, \qquad \sum_{e=(v,v') \in E} x_e^h - \sum_{e=(v',v) \in E} x_e^h \leq 0$$

- Type 8 constraints. Battery is only consumed on legs that are travelled.

$$\forall h \in H, \forall e \in E, \qquad z_e^h \leq x_e^h maxcharge(h)$$

- Type 9 constraints. The amount of battery left after leaving a request is equal to the amount of battery when arriving at that request less the amount of battery consumed in leaving the request.

$$\forall h \in H, \forall r \in R, \qquad \sum_{e=(v',v) \in E} z_e^h - \sum_{e=(v,v') \in E} z_e^h = \sum_{e=(v,v') \in E} x_e^h |e|$$

where v is the location of request r.

- Type 10 constraints. The number of legs remaining in a route only makes sense for edges in the route.

$$\forall h \in H, e \in E, \qquad t_e^h \leq x_e^h (2|R| + |V|)$$

Here, $2|R| + |V|$ was chosen as an upper bound for the number of legs in a minimal route, effectively leaving t_e^h unbounded wherever x_e^h is positive.

- Type 11 constraints. The number of legs remaining after departing a node (other than a starting node) is equal to the number of legs remaining upon arriving at that node less the number of departures from that node.

$$\forall v \in V \cup R, h \in H, \qquad \sum_{e=(v',v) \in E} t_e^h - \sum_{e=(v,v') \in E} t_e^h = \sum_{e=(v,v') \in E} x_e^h$$

where v is the location of request r. Although non-obvious, this is valid even if the quadcopter visits a node multiple times. The first arrival and last departure have meaningful values although any intermediate circuits may have shifted values.

- Type 12 constraints. The total time taken is at least as long as the time taken by each quadcopter. (this is how we fake a max() constraint)

$$\forall h \in H, \qquad \sum_{e \in E} x_e^h |e| \leq T$$

For the actual implementation, variables are arranged into NxN blocks, corresponding to the edge index. Within each block, the nodes are arranged in the order, request locations, quadcopter start locations, then ground vehicle locations. The variables are placed in column-major order within each block. The blocks are then arranged in order with x values first, followed by q's, then z's, then t's.

As an example, suppose we are dealing with a single quadcopter, single request, single item, and single ground vehicle. The request will then be at location 1, the quadcopter at location 2, and the ground vehicle at location 3. The variables in the linear program will be arranged in the order,

$$x_{11}^1, x_{21}^1, x_{31}^1, x_{12}^1, x_{22}^1, x_{32}^1, x_{31}^1, x_{32}^1, x_{33}^1,$$
$$q_{11}^1, q_{21}^1, q_{31}^1, q_{12}^1, q_{22}^1, q_{32}^1, q_{31}^1, q_{32}^1, q_{33}^1,$$
$$z_{11}, z_{21}, z_{31}, z_{12}, z_{22}, z_{32}, z_{31}, z_{32}, z_{33},$$
$$t_{11}^1, t_{21}^1, t_{31}^1, t_{12}^1, t_{22}^1, t_{32}^1, t_{31}^1, t_{32}^1, t_{33}^1, \mathrm{T}$$

We can also make a number of preliminary cuts. These are inequalities that follow from the integrality constraints, but not from the linear constraints. Including these cuts isn't necessary, but they do constrain the linear relaxation, which can reduce solution time.

- Additional bounds. The legs departing from a quadcopter's starting location can be constrained based on cargo. If the end of the leg is a delivery and the quadcopter's cargo does not include the goods to be delivered, then that leg cannot be taken. Similarly, if the end of the leg is a pickup and there

is insufficient room remaining in the quadcopter's capacity, the leg cannot be taken.

- Type 13 constraints. All quadcopters must move. If a quadcopter is already at a vehicle we want it to wait, but this can be satisfied by moving the 0 distance to the vehicle.

$$\forall h \in H, \sum_{v' \in N} x^{h}_{start(h),v'} \geq 1$$

4 Simulation and Results

MATLAB is used to simulate the delivery quadrotor.

4.1 Model Simulation

The quadrotor model simulation shown in figure 6.

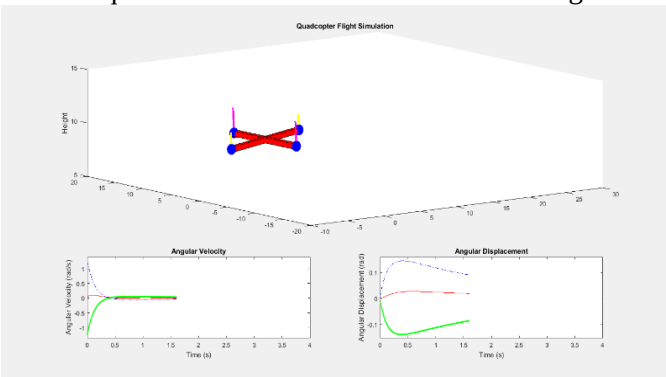

Figure 6: Model Simulation

4.2 Simulink Models

The followings are snapshots of the Simulink models

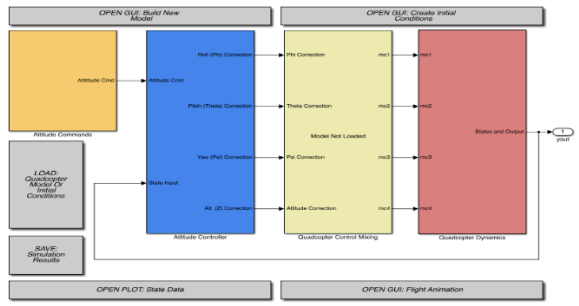

Figure 7: Attitude control Simulink model

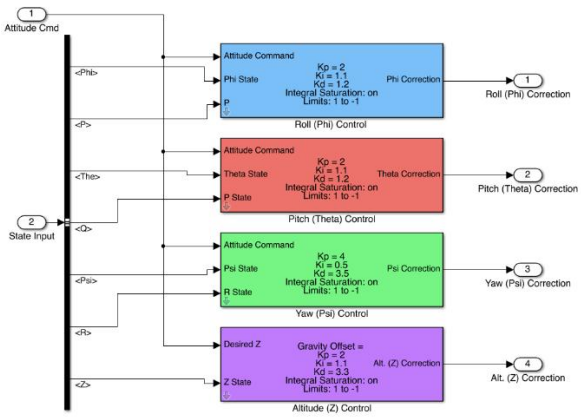

Figure 8: Attitude controller

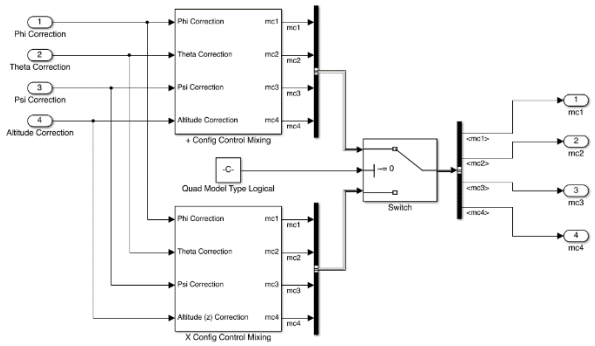

Figure 9: Quadcopter control mixing overview

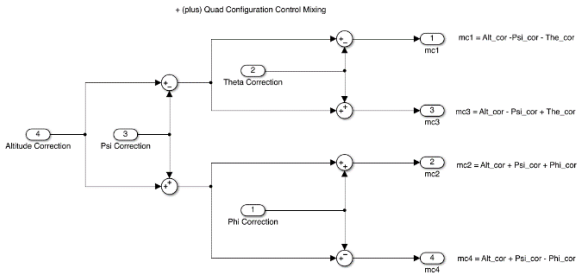

Figure 10: Plus configuration control mixing

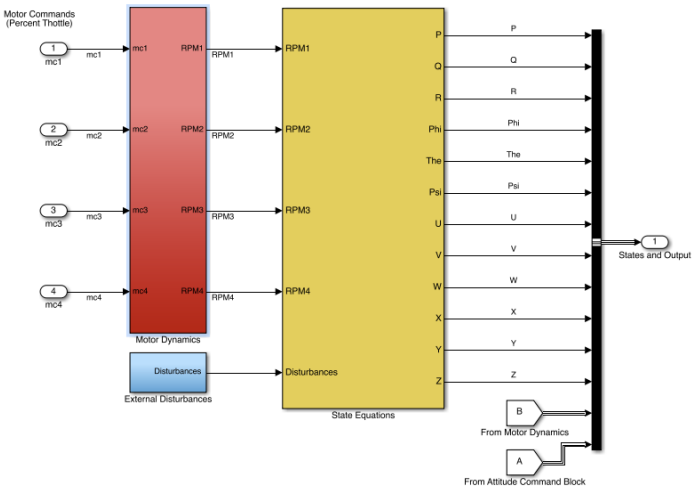

Figure 11: Quadcopter dynamic block

4.3 Attitude and Position Control (AC, PC) Results

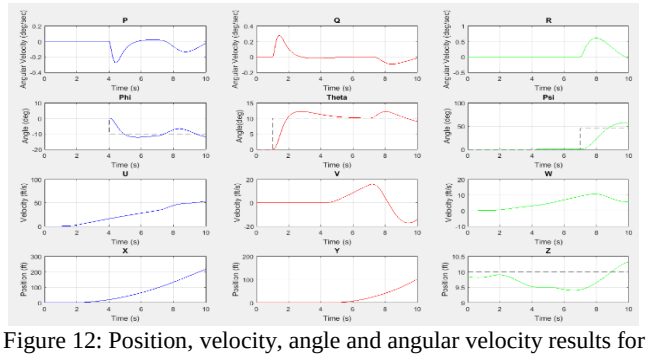

Figure 12: Position, velocity, angle and angular velocity results for AC

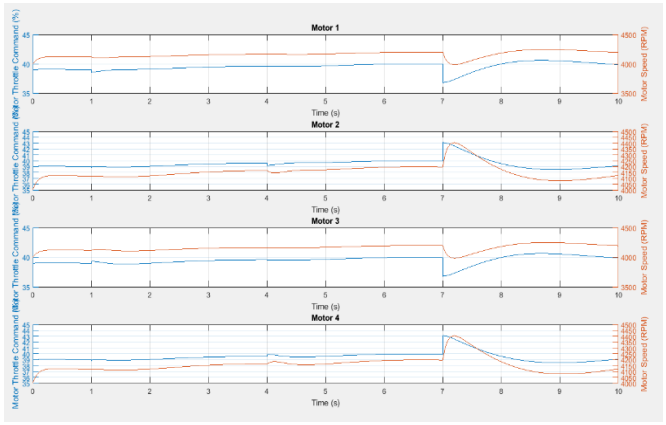

Figure 13: Motor results for AC

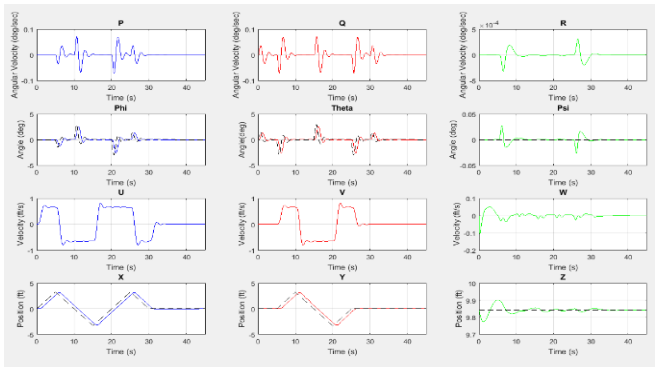

Figure 14: Position, velocity, angle and angular velocity results for PC

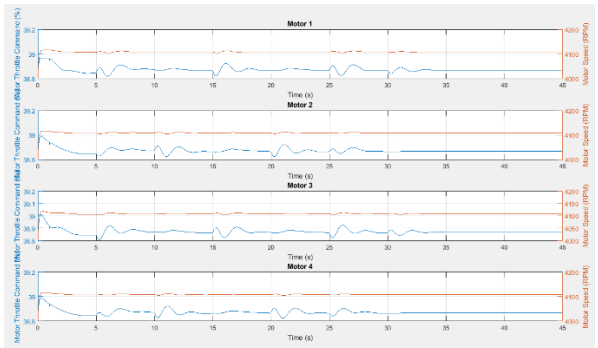

Figure 15: Motor results for PC

4.4 Computer Vision

The quadrocopter registers pictures, navigation data and videos and uploads them promptly to be handled by first getting the edge using Canny edge detection [9, 10] as shown in figure 16. Then by applying data fusion, the system is able to navigate well by keeping the exact distant on its way around the walls.

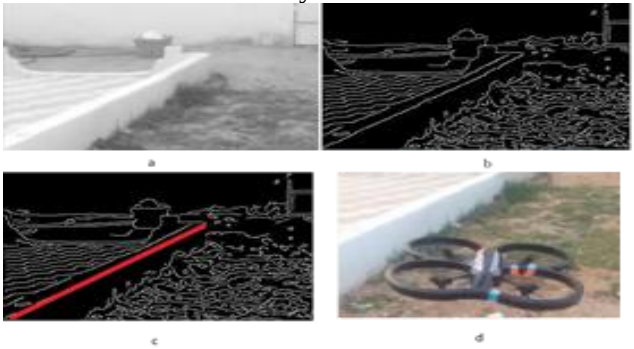

Figure 16: Canny edge detection

The quadrotor updates its position on a local map as shown in the following section.

4.5 Autonomous "SLAM"

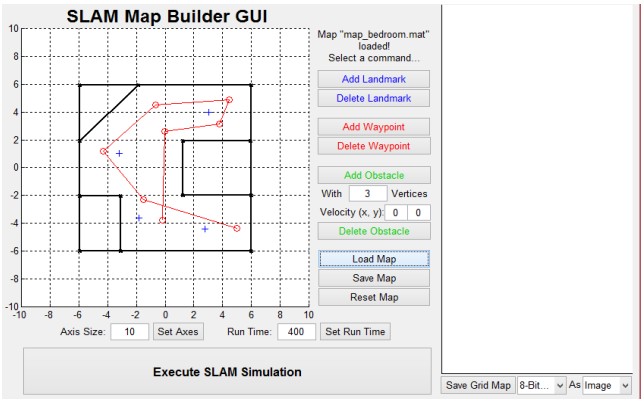

Figure 17: SLAM Map Builder

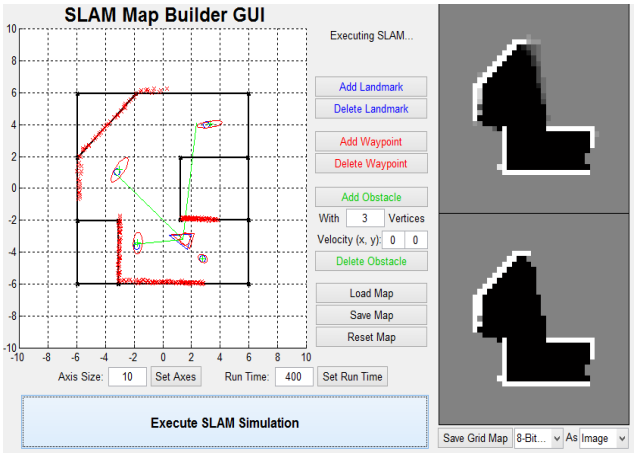

Figure 18: SLAM in progress

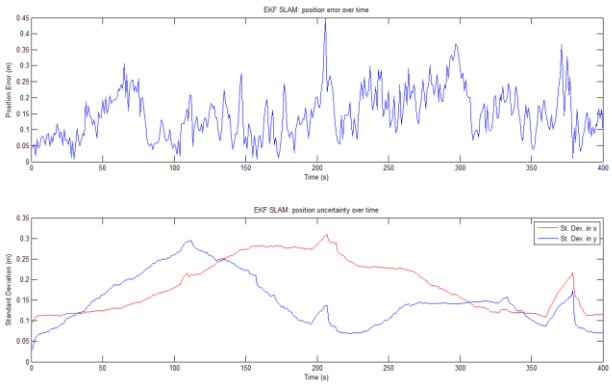

Figure 19: Position error and uncertainty for SLAM

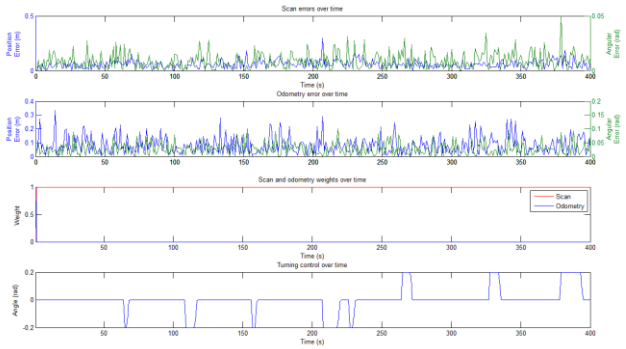

Figure 20: SLAM results

5 Conclusion and future work

The aim of this paper is to present the specification and the implementation of an autonomous quadrotor delivery system that fuses data from camera and IMU (Inertial Measurement Unit) to allow for high precision visual-inertial navigation. In this paper, an autonomous quadrotor delivery system is proposed. The specification of the system is proposed. The implementation of the system introduced SolidWorks design for the system, mathematical modelling for the pickup and delivery problem (PDP). Simulation results using MATLAB have been shown for the implementation of the model, control system and simultaneous localization and mapping (SLAM) using Extended Kalman Filter algorithm (EKF-SLAM) used for making the system autonomous. Experimental results for the use of canny edge detection algorithm have been shown. Future work include using solar energy for charging and make a collaboration among quadrotors as a swarm.

References

[1] Lozano, Rogelio, ed. Unmanned aerial vehicles: Embedded control. John Wiley & Sons, 2013.

[2] Hou, Hongning, et al. "A simple controller of minisize quad-rotor vehicle." Mechatronics and Automation (ICMA), 2010 International Conference on. IEEE, 2010.

[3] Kim, Jinhyun, Min-Sung Kang, and Sangdeok Park. "Accurate modeling and robust hovering control for a Quad–rotor VTOL aircraft." Journal of Intelligent and Robotic Systems 57.1-4 (2010): 9.

[4] Mourikis, Anastasios I., et al. "Vision-aided inertial navigation for spacecraft entry, descent, and landing." IEEE Transactions on Robotics 25.2 (2009): 264-280.

[5] Kottas, Dimitrios G., et al. "On the consistency of vision-aided inertial navigation." Experimental Robotics. Springer International Publishing, 2013.

[6] Mosterman, Pieter J., et al. "A heterogeneous fleet of vehicles for automated humanitarian missions." Computing in Science & Engineering 16.3 (2014): 90-95.

[7] Hartman, M. M. S. M. D., K. Landis, and J. K. D. B. C. Chang. "Quadcopter dynamic modeling and simulation." freeware project presented at "2014 MATLAB and Simulink Student Design Challenge". 2015.

[8] Durrant-Whyte, Hugh, and Tim Bailey. "Simultaneous localization and mapping: part I." IEEE robotics & automation magazine 13.2 (2006): 99-110.

[9] Green, Bill. "Canny Edge Detection Tutorial.", (2002).

[10] Bailey, Mark Willis. Unmanned aerial vehicle path planning and image processing for orthoimagery and digital surface model generation. Diss. Vanderbilt University, 2012.

[11] D'Angelo, R., and R. Levin. "Design of an Autonomous Quad-rotor UAV for Urban Search and Rescue." Worcester Polytechnic Institute (2011).

[12] Bouabdallah, Samir, Andre Noth, and Roland Siegwart. "PID vs LQ control techniques applied to an indoor micro quadrotor." Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on. Vol. 3. IEEE, (2004).

[13] Bresciani, Tammaso. "Modelling, identification and control of a quadrotor helicopter." MSc Theses, Department of Automatic Control, Lund University (2008).

[14] Bouabdallah, Samir, Pierpaolo Murrieri, and Roland Siegwart. "Design and control of an indoor micro quadrotor." Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on. Vol. 5. IEEE, (2004).

[15] Johnson, Eric N., and Michael A. Turbe. "Modeling, control, and flight testing of a small ducted-fan aircraft." Journal of Guidance Control and Dynamics 29.4 (2006): 769-779.

[16] Hoffmann, Gabriel M., et al. "Quadrotor helicopter flight dynamics and control: Theory and experiment." Proc. of the AIAA Guidance, Navigation, and Control Conference. Vol. 2. (2007).

[17] Gibiansky, Andrew. "Quadcopter dynamics, simulation, and control." Andrew Gibiansky:: Math→[Code] 21 (2012).

[18] Siegwart, Roland, Illah Reza Nourbakhsh, and Davide Scaramuzza. Introduction to autonomous mobile robots. MIT press, (2011).

[19] Thrun, Sebastian, Wolfram Burgard, and Dieter Fox. *Probabilistic robotics*. MIT press, 2005.

[20] Crowley, James L. "World modeling and position estimation for a mobile robot using ultrasonic ranging." Robotics and Automation, 1989. Proceedings., 1989 IEEE International Conference on. IEEE, 1989.

[21] Ayache, Nicholas, and Olivier D. Faugeras. "Building, registrating, and fusing noisy visual maps." The International Journal of Robotics Research 7.6 (1988): 45-65.

[22] Jordan, M., "An Introduction to Probabilistic Graphical Models", University of California, Berkeley, 2003.

[23] Cheeseman, P., R. Smith, and M. Self. "A stochastic map for uncertain spatial relationships." 4th International Symposium on Robotic Research. 1987.

[24] Bar-Shalom, Yaakov. *Tracking and data association*. Academic Press Professional, Inc., 1987.

[25] Neira, José, and Juan D. Tardós. "Data association in stochastic mapping using the joint compatibility test." IEEE Transactions on robotics and automation 17.6 (2001): 890-897.

[26] Stuart Russell and Peter Norvig, "Artificial Intelligence: A Modern Approach," Prentice-Hall, Englewood Cliffs, NJ, 2010,3rd edition.