自律飛行を目的としたドローンの飛行制御

2017SC029 小森脩平　2017SC058 岡田佳樹
2017SC088 山田樹
指導教員：中島明　坂本登

1　はじめに

近年小型ドローンの普及が進んでおり，様々な用途で使用されている．ドローンを手動で操縦する際機体を視認できない場合もしくは視認しにくい場合，操縦者はドローンの位置や姿勢が把握できず操縦が難しくなる．そのような場合に一定の経路の連続的な飛行や自動的な帰還などを行うために，何らかの指標を基にした自律飛行能力が必要となる．

現在の自律飛行は GPS で自らの位置を把握しながら指定された場所に移動させる手段が一般である．しかし，トンネル中や屋内などの GPS 電波の受信が不安定な場合，また定められた場所に高精度で接近，ホバリングする場合は非 GPS による誘導方式が必要となる．そこで本研究では屋内での運用と仮定し GPS を用いないドローンの誘導システムに AR マーカーを使用することを考える．

AR マーカーとは，現実世界と情報世界を重ね合わせて表示する AR システムで利用する画像である．自律飛行の誘導方式として考えられる GPS やモーションキャプチャ等に比べ低コストで用意がしやすい，GPS が届かない屋内での使用が可能，1 つのマーカーに与えられる情報量が多い等の利点がある．そのため本研究ではドローンの自律飛行に AR マーカーによる誘導方式を採用する．

上記のドローンの自律飛行を実現させるためには，ドローンが AR マーカーを認識可能な位置での一定時間のホバリングが必要であり，そのためにドローンの姿勢安定化制御が重要となる．次章からはドローンの姿勢安定化のためのモデリングと AR マーカーをドローンに認識させるシステムの実装，姿勢角制御器及び高度制御器の作成，そして，目標姿勢及び目標高度に追従させることを目的とした制御を行う．

2　ドローンのモデリング

2.1　ドローンの座標系とパラメータ

3 次元空間にあるドローンの空間表現を行う際，位置と姿勢角が必要になる．これらの状態量を表現するため，基準となる直交座標系である基準座標系 (Σ_r)，ドローンに固定された機体座標系 (Σ_b) という二つの直交座標系の定義を行う．これら二つの直交座標系はともに右手座標系である．また文字の左上添え字は基準となる座標系，右下添え字は表現される座標系を示しており，b は機体座標系，w は基準座標系を示す．以下の表 1 にドローンの状態パラメータを，図 1 にドローンの座標系を示す．また，U_f を以下の式 (1) のように定義する．

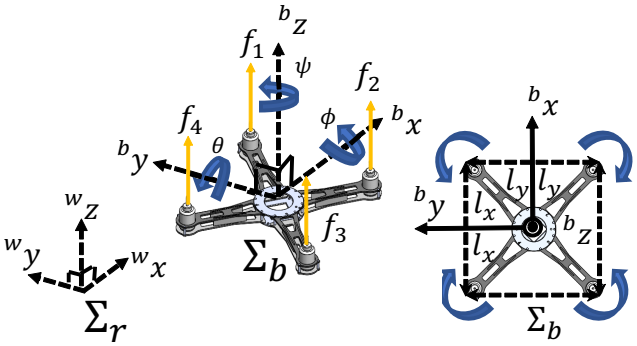


図 1　ドローンの座標系

$$U_f = \sum_{i=1}^{4} f_i \tag{1}$$

表 1　ドローンのモデリングに関する各種パラメータの定義

記号	名称及び単位
m_b	機体の質量 [kg]
x	機体の x 方向への位置座標 [m]
y	機体の y 方向への位置座標 [m]
z	機体の z 方向への位置座標 [m]
ϕ	機体の姿勢角 (roll 角)[rad]
θ	機体の姿勢角 (pitch 角)[rad]
ψ	機体の姿勢角 (yaw 角)[rad]
J_{xx}	機体の x 軸慣性モーメント [kgm^2]
J_{yy}	機体の y 軸慣性モーメント [kgm^2]
J_{zz}	機体の z 軸慣性モーメント [kgm^2]
l_x	ロータと y 軸間の距離 [m]
l_y	ロータと x 軸間の距離 [m]
f_i	ロータ i 番目の推力 [N]

2.2　ドローンモデルの導出

q を基準座標系から見たドローンの位置，ドローンの姿勢角を含む一般化座標とする．ドローンの位置ベクトルを $^wP_b = [x,\ y,\ z]^T$，ドローンの姿勢角を $\eta = [\phi, \theta, \psi]^T$ とすると

$$q = [^wP_b^T\ \eta^T]^T \in \mathbb{R}^6 \tag{2}$$

を得ることができる．次に ω を回転速度ベクトルとし，運動エネルギー $T_b(q, \dot{q})$ とポテンシャルエネルギー $U_b(q)$ からラグランジュ関数 L_b を以下のように表すことができる．

$$L_b(q, \dot{q}) = T_b(q, \dot{q}) - U_b(q) \tag{3}$$

$$T_b(q,\dot{q}) = \frac{1}{2}(m_b^w \dot{P}_b^T \ {}^w\dot{P}_b + w^T \ {}^bJ_b w) \tag{4}$$

$$U_b(q) = m_b g^w e_z^T \ {}^wP_b \tag{5}$$

この時 $e_z = [0,0,1]^T$ は各座標系での z 軸方向の単位ベクトルであり，${}^bJ_b$ =diag(J_{xx}, J_{yy}, J_{zz}) である．

2.3 ラグランジュの運動方程式から状態方程式の導出

並進運動では力，回転運動ではモーメントのことを指す一般化力を $F = B_f u$ とおくとラグランジュの運動方程式は式 (3) を用いて以下の式 (6) となる．

$$\frac{d}{dt}\left(\frac{\partial L_b(q,\dot{q})}{\partial \dot{q}}\right) - \frac{\partial L_b(q,\dot{q})}{\partial q} = B_f u \tag{6}$$

右辺の $B_f u$ について，ロータの推力 u を $u = [f_1, f_2, f_3, f_4]^T$ とし，B_f を一般化力から u 以外の要素をまとめたものとする．さらに運動エネルギーは慣性行列 $M(q)$ を用いて以下のように表せる．

$$T_b(q,\dot{q}) = \frac{1}{2}\dot{q}^T M(q)\dot{q} \tag{7}$$

式 (6) と式 (7) より

$$M(q)\ddot{q} + \frac{d}{dt}(M(q))\dot{q} - \frac{\partial T_b(q,\dot{q})}{\partial q} + \frac{\partial U_b(q)}{\partial q} = B_f u \tag{8}$$

が得られ，ここで

$$N(q,\dot{q}) = \frac{d}{dt}(M(q))\dot{q} - \frac{\partial T_b(q,\dot{q})}{\partial q} + \frac{\partial U_b(q)}{\partial q} \tag{9}$$

と置くことにより式 (6) は

$$M(q)\ddot{q} + N(q,\dot{q}) = B_f u \tag{10}$$

と変形できる．式 (10) は

$$\begin{aligned} M(q)\ddot{q} &= -N(q,\dot{q}) + B_f u \\ \Leftrightarrow \ddot{q} &= -M^{-1}(q)N(q,\dot{q}) + M^{-1}(q)B_f u \end{aligned} \tag{11}$$

となり，状態変数 X を $X = [q^T, \dot{q}^T]^T$ として非線形状態方程式の

$$\dot{X} = f(X) + g(X)u \tag{12}$$

を得ることができる．ただし，

$$f(X) = \begin{bmatrix} \dot{q} \\ -M^{-1}(q)(N(q,\dot{q})) \end{bmatrix},\ g(X) = \begin{bmatrix} O_{6\times 4} \\ M^{-1}(q)B_f \end{bmatrix} \tag{13}$$

である．[1]

3 制御則設計

3.1 制御器設計に用いる変数

制御器設計を行う上で用いる変数の定義を表 2 に示す．この節ではドローンの制御器設計を行う．ドローンの高度及び姿勢角の目標値追従を行うために高度制御及び姿勢角制御では PID 制御を採用した．以下の図 2 に制御系の詳細を示す．

表 2 制御に関するパラメータの定義

記号	名称
ϕ	現在の roll 角
θ	現在の pitch 角
ψ	現在の yaw 角
ϕ_{ref}	roll 角の目標値
θ_{ref}	pitch 角の目標値
ψ_{ref}	yaw 角の目標値
Z	現在の高度
Z_{ref}	高度の目標値

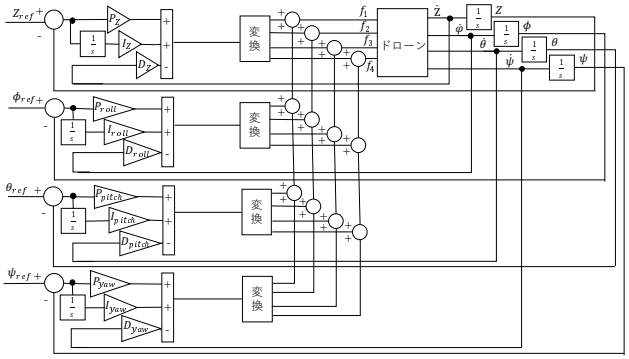


図 2 制御系の詳細

3.2 高度制御の詳細

ドローンの PID 制御に関するパラメータを表 3 に示す．

表 3 ドローンの PID 制御に関するパラメータ

記号	名称
P_{roll}	roll 角度制御の P ゲイン
P_{pitch}	pitch 角度制御の P ゲイン
P_{yaw}	yaw 角度制御の P ゲイン
P_z	高度制御の P ゲイン
I_{roll}	roll 角度制御の I ゲイン
I_{pitch}	pitch 角度制御の I ゲイン
I_{yaw}	yaw 角度制御の I ゲイン
I_z	高度制御の I ゲイン
D_{roll}	roll 角度制御の D ゲイン
D_{pitch}	pitch 角度制御の D ゲイン
D_{yaw}	yaw 角度制御の D ゲイン
D_z	高度制御の D ゲイン

高度制御に関して微分先行型 PID 制御を用いて制御器の設計を行う．高度に関する運動は回転運動に影響を及ぼさない．z 方向の並進運動の入力を U_z とすると z 方向の

入力は以下のようになる．

$$U_z = m_b + P_z(z_{ref} - z) + I_z \int_0^t (z_{ref} - z)d\tau - D_z\dot{z} \quad (14)$$

ここで高度の PI 制御及び速度のフィードバックにより生じる推力は式 (1) より

$$U_z = \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix} \quad (15)$$

となる．したがって高度制御のために必要な各ロータの推力への分配を行う変換は以下のようになる．

$$\begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix} = \begin{bmatrix} \frac{1}{4} \\ \frac{1}{4} \\ \frac{1}{4} \\ \frac{1}{4} \end{bmatrix} U_z \quad (16)$$

3.3 姿勢制御の詳細

姿勢制御に関しても微分先行型 PID 制御を用いて制御器の設計を行う．微分先行型 PID 制御を用いた結果，入力は式 (17) となる．ただし $e_\phi = \phi_{ref} - \phi$，$e_\theta = \theta_{ref} - \theta$，$e_\psi = \psi_{ref} - \psi$ である．

$$\begin{bmatrix} \tau_x \\ \tau_y \\ \tau_z \end{bmatrix} = \begin{bmatrix} P_{roll}e_\phi + I_{roll}\int_0^t e_{w\phi}d\tau - D_{roll}\dot{\phi} \\ P_{pitch}e_\theta + I_{pitch}\int_0^t e_{w\theta}d\tau - D_{pitch}\dot{\theta} \\ P_{yaw}e_\psi + I_{yaw}\int_0^t e_\psi d\tau - D_{yaw}\dot{\psi} \end{bmatrix} \quad (17)$$

ここで roll 角の PID 制御により生じたトルク τ_x は以下のようになる．

$$\tau_x = L_x \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix}, L_x = \begin{bmatrix} l_y & -l_y & -l_y & l_y \end{bmatrix} \quad (18)$$

よって roll 方向に回転させるために必要なトルクから各ロータの推力への変換は

$$\begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix} = L_x^T(L_xL_x^T)^{-1}\tau_x \quad (19)$$

$$\begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix} = \begin{bmatrix} \frac{1}{4l_y} \\ -\frac{1}{4l_y} \\ -\frac{1}{4l_y} \\ \frac{1}{4l_y} \end{bmatrix} \tau_x \quad (20)$$

となる．τ_y，τ_z に関するモーメントから各ロータへの推力の変換も同様に行う．

4 シミュレーション

前節で説明した制御器についてのシミュレーションを行う．

4.1 シミュレーションの状況設定

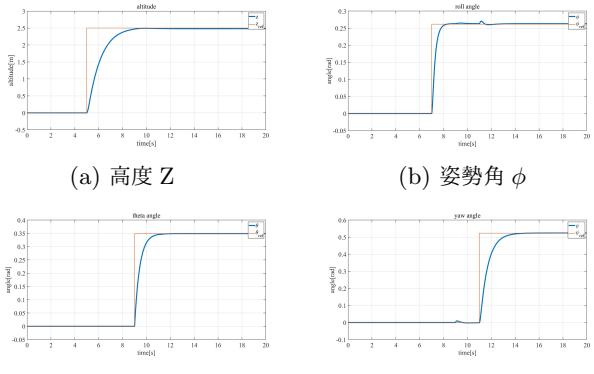
シミュレーションの状況設定は，ドローンをホバリングさせ roll 角と pitch 角に姿勢角目標値を与え時間をずらし高度目標値を与える，とした．ドローンの初期値は位置，姿勢角，速度，加速度全て 0 であるとする．目標値は高度 2.5[m]，姿勢角に $\phi = \frac{\pi}{12}$[rad]，$\theta = \frac{\pi}{9}$[rad]，$\psi = \frac{\pi}{6}$[rad] とした．また，高度 Z には 5[s]，姿勢角 ϕ には 7[s]，θ には 9[s]，ψ には 11[s] で目標値を印加した．設定した PID ゲインを以下の表に記載する．

表 4 PID ゲイン

記号	名称	ゲイン
P_{roll}	roll 角度制御の P ゲイン	0.5
P_{pitch}	pitch 角度制御の P ゲイン	3
P_{yaw}	yaw 角度制御の P ゲイン	0.5
P_z	高度制御の P ゲイン	13
I_{roll}	roll 角度制御の I ゲイン	0.01
I_{pitch}	pitch 角度制御の I ゲイン	0.001
I_{yaw}	yaw 角度制御の I ゲイン	0.001
I_z	高度制御の I ゲイン	0.3
D_{roll}	roll 角度制御の D ゲイン	0.15
D_{pitch}	pitch 角度制御の D ゲイン	1.3
D_{yaw}	yaw 角度制御の D ゲイン	0.35
D_z	高度制御の D ゲイン	15

4.2 シミュレーション結果と考察

高度と姿勢角のシミュレーション結果を以下の図 3 に示す．



(a) 高度 Z　(b) 姿勢角 ϕ

(c) 姿勢角 θ　(d) 姿勢角 ψ

図 3 シミュレーション結果

図 3 より，姿勢角 ϕ のグラフの 11[s] 辺りの応答に見られるように，各姿勢角のカップリングが見られるが，設計した制御則では，目標値に対する追従性が十分であることが見てとれる．

5 AR マーカー

5.1 AR とは

AR は「Augmented Reality」の略称であり現実環境に情報を付加提示する技術，および情報を付加提示された環境を指す．AR には『ロケーションベース AR』,『マーカー型ビジョンベース AR』,『マーカーレス型ビジョンベース AR』[2] という 3 種類の代表的な型が存在する．

ロケーションベース AR は GPS に基づく位置情報に付加的情報を表示する技術である．本研究の GPS を使用できない場所での運用には適さないため採用していない．

ビジョンベース AR は特定の物体や図形上に付加情報を表示する技術である．AR マーカーを標識としてドローンの誘導が可能であると考えられるため，マーカー型ビジョンベース AR を本研究では採用する．

AR マーカーとはマーカー型ビジョンベース AR に用いられる画像であり，正方形の黒枠に囲まれた白黒のパターンで構成される．黒枠でマーカーの検出を行い，枠の内側のパターンによってマーカーの ID を判別することができる．またマーカーは ID の情報の他に座標軸を持っており，座標軸によって認識機器との距離角度を測定することができる．マーカーの ID によってどの情報を表示するかを決定し，座標軸によって画像内のどの位置にどの向きで情報を表示するかを決定することで高精度の情報の表示が可能な技術と言える．[3][4][5]

本研究では AR マーカーを用いることでドローンの飛行の様子を決定し，マーカーとドローンの位置関係を利用し移動先の位置を決定することでドローンの誘導を行う．

5.2 AR 実装用デバイスの選定

ドローンに搭載したカメラが撮影した映像を認識させ飛行指令を出すにあたり，映像を撮りその画像を認識し処理した結果に応じて指令を出すシステムが必要になる．従来のドローンに搭載している myRIO のみにこのシステムを実装し運用することが困難であるため，画像処理を行い指令を送る役割を担うシステムを実装するデバイスを用意する必要がある．本研究ではそのシステムを実現するにあたって Raspberry pi を使用する．Raspberry pi は AR マーカーを認識できるライブラリである ArUco を含む OpenCV を運用することができる．

5.3 使用機器

本研究では AR マーカーを認識するデバイスとして Raspberry pi zero W を使用する．Raspberry pi は ARM プロセッサを搭載したシングルボードコンピュータである．その中で Raspberry pi zero W は小型の種類であり，大きさ 65 × 30[mm] 質量 9[g] と他のモデルに比べ半分以下の質量となっている．小型軽量化・低コスト化が実現されている Raspberry pi zero W はドローン等の小型機への搭載にも実用的なデバイスである．これが本研究にこのデバイスを使用する理由の 1 つである．

Raspberry pi に接続し AR マーカーを撮影するカメラには For Raspberry pi カメラモジュール　5MP を使用する．固定焦点レンズに OV5647 ウェブカメラセンサーを搭載した 5 メガピクセルセンサーを持ち，2592 × 1944[pixel] の静止画，1080p/30fps，720p/60fps，および 480p/90fps のビデオ録画も可能である．

6 距離計測

Raspberry pi が myRIO に出す指令を切り替える際，カメラと AR マーカーの距離を計測し，その値を用いる．詳細は 7 章にて説明を行うため，本章では Raspberry pi とカメラを用いた距離計測を当システムに実装するための検証を行っていく．

6.1 距離計測と角度計測

AR マーカーとドローンの位置，角度によりドローンの移動する位置を決定する上でカメラと AR マーカーで同様の数値を取得する必要がある．カメラで撮影した画像内の点と 3 次元空間に存在する 3 次元点の関係は以下の式 (21) で表される．

$$Z_c \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X_m \\ Y_m \\ Z_m \\ 1 \end{bmatrix} \tag{21}$$

式 (21) は画像内での位置を表す画像座標点 $[u, v, 1]^T$ とマーカーから見てある 3 次元点がどの位置にあるかを表すマーカー座標点 $[X_m, Y_m, Z_m]^T$，カメラから見た位置を表すカメラ座標点の Z 座標 Z_c，右辺の第 1 項の行列で表されるカメラの内部パラメータと第 2 項の行列で表される外部パラメータで構成される．この式により撮影した画像からカメラと被写体との距離と角度を測定することができる．

6.1.1 カメラの内部パラメータ

カメラから見た位置を表すカメラ座標点 $[X_c, Y_c, Z_c]^T$ には画像には存在しない奥行きである Z_c が存在する．そのためカメラ座標点を正規化し $Z_c = 1$ の平面に移動させる．このカメラ座標点に焦点距離 f を掛けることでカメラ座標点を画像平面上に置くことができる．その後画像の左上を原点とするためカメラ座標点を x 軸方向に c_x，y 軸方向に c_y 並進運動させる．これを数式で表現すると以下の式 (22) になる．

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{X_c}{Z_c} \\ \frac{Y_c}{Z_c} \\ 1 \end{bmatrix} \tag{22}$$

式 (22) を変形することで以下の式 (23) を得る．

$$Z_c \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} \tag{23}$$

式 (23) のカメラ座標点に掛かる行列が内部パラメータである．内部パラメータによって画像座標系の点である画像座標点をカメラ座標系のものに変換することができる．[6]

6.2　カメラの外部パラメータ

マーカー座標点をカメラ座標系で表現することでカメラから見た 3 次元点の向きを求めることができる．マーカー座標点をマーカー座標系とカメラ座標系の向きを揃えるよう回転移動させ，二つの座標系の原点を揃えるようマーカー座標系を並進移動させる．

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} X_m \\ Y_m \\ Z_m \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix} \quad (24)$$

式 (24) を同次座標により変形することで式 (25) の形になる．

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X_m \\ Y_m \\ Z_m \\ 1 \end{bmatrix} \quad (25)$$

式 (25) のマーカー座標点に掛かる行列が外部パラメータである．外部パラメータによってカメラ座標系の点であるカメラ座標点をマーカー座標系のものに変換することができる．[6]

6.3　カメラの解像度と距離計測の精度と処理速度

距離計測のプログラムはドローンで用いるには処理速度が遅いため, 高速化を図る必要がある. その手段として，画像のサイズを圧縮する方法を用いた. まず，カメラと AR マーカーの計測距離と，処理速度を表示させるプログラムを作成した．一辺が 5[cm] の AR マーカーをカメラから 0.3[m] の距離に配置し，プログラムを実行した．各画像サイズの結果を表 5 に示す. ただし，元の画像サイズは 640 × 480[pixel] であり，処理速度はプログラムが 1 ループするのにかかる時間を指すものとする．

表 5　実値が 0.3m 時の各画像サイズの計測距離と処理速度

画像サイズ [pixel]	平均測定値 [m]	誤差 [m]	処理速度 [s]
320 × 240	0.4819	0.1819	0.09
360 × 270	0.5037	0.2037	0.14
400 × 300	0.4433	0.1433	0.18
440 × 330	0.3730	0.0730	0.21
480 × 360	0.3852	0.0852	0.24
520 × 390	0.3381	0.0381	0.28
560 × 420	0.3089	0.0089	0.33
600 × 450	0.3071	0.0071	0.37
640 × 480	0.2859	0.0105	0.30

結果より，画像サイズを圧縮するほど処理速度が短縮されるが，圧縮すればするほど誤差も大きくなる傾向があることが分かる．距離の計測誤差と処理速度の兼ね合いを考慮した結果，画像サイズを 440 × 330[pixel] に圧縮して以降のプログラムを作成していくことを決定した．

6.4　距離計測を用いた指令切り替え

7 章の図 6 のフローチャートに沿い，プログラムを作成した．台座に Raspberry pi zero W とカメラ載せてドローンに搭載した状況を模擬し，想定したタイミングで指令が切り替わるかの検証を行った．結果として，想定したタイミングで指令が切り替わる事が確認出来た．指令値の調整を実機実験にて今後行っていく予定である．

7　ドローン飛行システム

本研究は以下の図 4 に示した手順を踏んでドローンの自律飛行を実現させることを目指す．

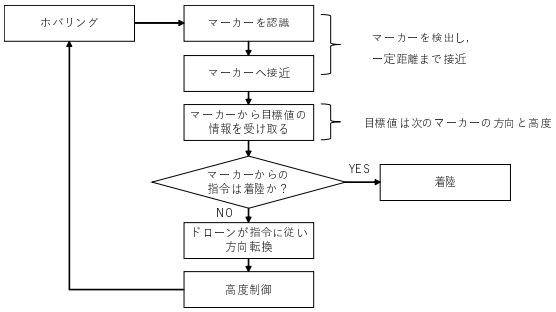


図 4　飛行システムの概要

上記の飛行を実現させるために，AR マーカーを撮影するカメラ，撮影した画像を処理する Raspberry pi，画像処理の結果によって変わる目標値を受け取りドローンへ制御入力を行う myRIO が必要となる．このシステム全体の構成を以下の図 5 に示し，Raspberry pi のシステムの概要を図 6 に示す．

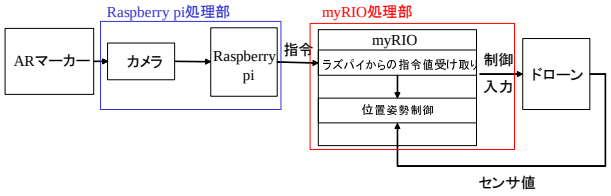


図 5　システム全体の構成図

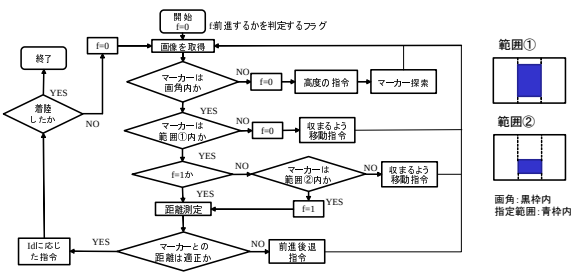


図 6　Raspberrypi 側のシステムの概要

8　実機実験

8.1　飛行時のマーカー認識

静的状態でのマーカーの認識の検証を行ったため，動的状態でのマーカーの認識の検証を行っていく．この実験を行う目的は，ドローンの飛行中にマーカーを認識させ AR マーカー認識システムをドローンに搭載した場合に正常に機能するかを検証するところにある．

8.1.1　実験内容

ドローンに Raspberry pi とカメラを搭載して飛行させる．Raspberry pi は ssh 接続したノートパソコンで操作を行い，マーカー認識のプログラムをドローン飛行中に実行する．マーカーを認識した際にその旨を伝える文字列をエディタに表示することでマーカーの認識が正しく行われているかの確認を行う．またプログラム実行中に撮影した映像によりドローン飛行中の動きを確認する．

8.1.2　実験結果と考察

実験によってマーカーが正しく認識されていることを確認できた．しかし，マーカーが画角内に収まっていても認識できない瞬間が存在した．要因として主に 2 点考えられる．Raspberry pi の処理速度が追い付いていない点と振動がマーカー認識を阻害していた点である．撮影中の動画を確認したところ，プログラムのループ周期と動画のフレームレートが大きく乖離していた．これは動画を保存するプログラムを用いるにあたって様々な制約があったためと考えられる．保存する画像サイズの変更ができないため，画像サイズ圧縮による処理の軽量化ができず，また，保存するということ自体にリソースが割かれており，処理自体が重くなってしまっていた．そのため，動画保存の代わりにログをとるようにし，適切な圧縮率を探す，また，マイコンの性能を上げる必要があると認識した．また，カメラの固定が甘い状態で実験を行ったところ，全くマーカーを認識しなかった．しかし，カメラの固定を強固にすることで認識状況が大幅に改善された．これはカメラに伝わる振動によって，マーカーの認識が阻害されていたと考えられ，機体自体の飛行安定性を向上させることでさらなる精度の向上が見込まれる．

8.2　飛行時の距離計測

次に飛行中のドローンに AR マーカーの撮影を行わせ，動的状態で距離計測が可能かの検証を行っていく．この実験を行う目的は，実装予定のプログラムでは計測距離に応じて指令値を切り替える想定であるため，動的状態で距離計測が可能であるかを検証するところにある．

8.2.1　実験内容

ドローンに Raspberry pi とカメラを搭載して飛行させる．Raspberry pi は ssh 接続したノートパソコンで操作を行い，マーカーの距離計測のプログラムをドローン飛行中に実行する．マーカーを認識した際にその旨を伝える文字列と計測した距離を表す数字をエディタに表示することでマーカーの認識が正しく行われているかの確認を行う．またプログラム実行中に撮影した映像によりドローン飛行中の動きを確認する．

8.2.2　実験結果と考察

実験により動的状態であってもドローンとマーカーの距離を計測できることが確認できた．しかし，前述の実験と同様にマーカーが画角内に収まっていても計測できない瞬間が存在した．要因も前述の実験と同様に Raspberry pi の処理速度が追い付いていないことと振動がマーカー認識を阻害しているためだと考えられる．

9　終わりに

本研究ではシミュレーション上で複数の制御器を作成し，飛行安定化のためのドローンの姿勢角と高度の制御を行った．また，カメラによる距離計測が可能であることを確認し，それに応じて指令を切り替えるプログラムを作成した．そして動的状態のドローンで距離計測が可能であることを確認した．

しかしデバイス間の通信の確立を行うことはできず，指令値も未調整であるため，今後の課題である．

また，画像処理高速化のために圧縮率の選定を行い，ドローンの飛行安定性を向上させることによって認識の精度向上を図り，AR マーカーを用いた非 GPS 下での自律飛行の実現を目指す．

参考文献

[1] 林美咲，宮野峻，西田裕貴，米川翔太：『クアッドコプターの飛行安定化制御システムの開発』．2018 年卒業論文，南山大学理工学部機械電子制御工学科坂本・中島研究室，2018.

[2]『いまさら聞けない AR（拡張現実）の基礎知識』．https://bit.ly/3lZe2QS，2011.

[3]『【誰でも分かる】AR(拡張現実) の仕組み・技術』．https://bit.ly/341LQqk，2017.

[4] 菊地慶仁，阿部太智．AR マーカーに基づくドローンの自律飛行 (第二報)．2018 年研究論文，北海学園大学工学部電子情報工学科，2019.

[5] 山田樹，田中秀幸，松本吉央．AR マーカと高精度マーカを融合したドローンの誘導着陸システムの開発．筑波大学，2019.

[6] Gary Bradski，Adrian Kaebler（松田晃一　訳）：詳解 OpenCV コンピュータビジョンライブラリを使った画像処理・認識．株式会社オライリージャパン，東京，2015