卒業論文 2019 年度 (令和元年度)

QR コードを用いたロボットナビゲーションシステムの開発

慶應義塾大学 環境情報学部環境情報学科
井手田悠希

卒業論文要旨 - 2019 年度 (令和元年度)

QR コードを用いたロボットナビゲーションシステムの開発

人間が物品管理を行っている倉庫で物品管理の自動化を行おうとする場合二通りの方法が考えられる．一つ目が倉庫内にレールを設置しレール上を専用機が移動しながら物品管理を行う方法．二つ目は倉庫内は何も手を加えずに巡回ロボットを用いて物品管理を行う方法である．本研究では 2 つ目の方法で物品管理に利用されることを想定したロボットのナビゲーションシステムの開発を目指す．ロボットには移動の自由度が高いドローン (小型無人航空機) を採用している．その上でロボットの経路情報の管理は非常に重要であり，現状ロボットのナビゲーションシステムでは経路情報はソフトウェアによる管理が主である．一方で管理対象の規模が大きくなる場合その経路情報も同時に変更になると考えられ，規模変更に合わせてシステム自体の修正が必要になる．そこで今回は管理対象の規模変更によるシステムの修正を減らす為に経路情報に関してステートレスなナビゲーションシステムを開発出来ればこれらのデメリットは解消すると考えた．本研究では QR コードを用いて経路情報に関してステートレスなナビゲーションシステムの実現をする．システム構築を行う上では ROS を採用し使用するロボットが変化した場合にも対応が容易となるような設計を行う．

キーワード:
1. ドローン, 2. ナビゲーションシステム, 3. 倉庫, 4. QR コード, 5. ROS,

慶應義塾大学 環境情報学部環境情報学科
井手田悠希

Abstract of Bachelor's Thesis - Academic Year 2020

Development of a robot navigation system with QR code

There is 2 way which realizes automate item management system in the warehouse. The first way is putting some rail and using dedicated robots on the rail. The second way is using patrol robots without any warehouse reform. This research focuses on the robot which is used in a second way. We use a drone as a robot because the drone has a high degree of freedom. Moreover, It is important that is route information and is managed by software mainly. On the other hand, we need to update or install navigation software when the route information or robot has changed. Furthermore, the software is going to large when routes increased. Thus, we propose a navigation system which is stateless of route information. In this system, we use QRCode to make it stateless of route information. We use ROS to build this system because making it easy to change the system when a robot is going to change.

Keywords :
1. Drone, 2. Navigation System, 3. WareHouse, 4. QRCode, 5. ROS,

Keio University Bachelor of Arts in Environment and Information Studies
Yuki Ideta

目 次

図 目 次

表 目 次

第1章　序論

本章では本研究の背景，課題及び手法を提示し，本研究の概要を示す．

1.1　本研究の背景

1.1.1　自己位置推定手法の発展

モバイルデバイスの高度化やセンサの小型化，計算能力の向上により高度な処理が可能となった．これにより自己位置推定の様々な手法が提案され，その精度も向上を続けている．しかしながら実運用を考慮する上で運用コストや機体性能，環境による制約が多く要件に応じた適切な自己位置推定手法を取る必要がある．

1.1.2　倉庫内での物品管理方法

現在倉庫内の物品管理[1]に用いられる方法はいくつかあり，自動型の物であれば物品棚自体に設置されたレール上を物品管理，物品の配置/取得する専用機が移動するという形が取られている．この形を基本形として取得後はベルトコンベアーで移動，梱包等まで自動で行う倉庫もある．一方で人間が専用機を用いて物品管理や物品の配置/取得をする形式の倉庫も多く存在している．現在人間が物品管理を行っている倉庫で新たにレールを用いた形で自動化を行おうとするとレールの敷設が必要となる．ここでレールの敷設が不要な自動物品管理の方法としてロボットを巡回させる形での物品管理の自動化が考えられる．また，この際用いられるロボットとしては自由度の高いドローンを用いるのが効率的である．そこで本研究では将来的に物品管理に利用されることを想定したドローンのナビゲーションシステム[2]の開発を目指す．

1.2　本研究の問題と仮説

現在自己位置推定にはGPS(Global Positioning System)を用いる手法が非常に有力であり頻繁に用いられる．しかし，屋内等の理由により直上方向に遮蔽物がある場合にはその精度が著しく低下することや利用できないという状況が発生する．その為屋内では後述

[1]本論文では定期的に物品の位置の記録を行うことを指す
[2]本論文ではロボットの自己位置推定と経路選択を行うシステムを指す

する Visual SLAM や LiDAR SLAM，他に TDOA を用いた自己位置推定手法が利用されている．

実運用を想定したロボットにおいて重要な要素としてメンテナンス性が挙げられる．

その一つ目が経路情報の管理である．倉庫内を複数機で巡回する形式のドローンの運用を考えるとその経路情報の管理はいくつか方法が考えられる．はじめに考えられるのは機体それぞれに飛行プログラムの一部として経路情報を書き込む方法である．外部との通信を必要とせず自律飛行性能は高くなるというメリットがあるが，常に最新の経路情報で飛行させようとすれば経路情報のバージョン管理や飛行前のアップデート確認の手間が必要となるというデメリットがある．次に考えられるのがグラウンドコントロールシステムによる中央集権的な管理である．基本の飛行システムさえ書き込んでおけば経路情報と誘導はグラウンドコントールシステム側で一元管理が可能になり飛行前のアップデートの確認の手間は不要になるというメリットがあるが．機体側は外部通信機構が必須になる上グラウンドコントロールシステム側での複数機体の管理，グラウンドコントロールシステムと各機体とのコミュニケーションが必要となる上に機体数が増えた場合にそのネットワークの帯域も問題となる可能性があるというデメリットがある．

二つ目が機体自体の管理である．上述したようなシステムでは機体の自己位置推定を行う上で専用の機材や画像から特徴点抽出及び環境地図の三次元再構成を行えるような計算リソースが必要となる．

また，現在これらの要件を満たすドローンを専用基板などを発注せずに OSS や各社より販売される開発用フライトコントローラ[3]を利用して自作しようとすれば，充分な積載量を確保するためには一辺 40cm 程の機体サイズとなる．倉庫のように人と同じ空間を飛行するドローンとしては少々恐怖感を与えるサイズとなる上事故時の被害もトイドローンのような 10cm 代の小型のものより大きくなる．

そこで今回は管理対象の規模変更によるシステムの修正を減らす為に経路情報に関してステートレスなナビゲーションシステムを開発出来ればこれらのデメリットは解消すると考えた．本研究では QR コードを用いて経路情報に関してステートレスなナビゲーションシステムの実現をする．

1.3 本論文の構成

本論文における以降の構成は次の通りである．

2 章では，本研究の要素技術となる PnP 問題と SLAM と QR コードに着いて整理する．

3 章では，本研究における問題の定義と，解決するための要件の整理を行う．

4 章では，本研究の提案手法を述べる．

5 章では， 4 章で述べたシステムの実装について述べる．

6 章では， 3 章で求められた課題に対しての評価を行い，考察する．

8 章では，本研究のまとめと今後の課題についてまとめる．

[3] ドローンが飛行する上で姿勢維持や飛行命令を行う装置

第2章　本研究の要素技術

本章では本研究の要素技術となる PnP 問題と GraphBasedSLAM と QR コードについて各々整理する．

2.1　PnP 問題

PnP(Perspective-n-Point) 問題は 3 次元空間上の n 個の点の情報が与えられた際にカメラ座標系に投影された 2 次元平面状の同じ点をを用いてカメラの外部パラメータ[1]を求める．という問題を指す．

本論文上の実装では QR コードの左上の角を世界座標系原点として推定を行う．QR コードの四角を利用し P4P 問題として帰着させる．

2.2　QR コード

自動車メーカーのデンソーに発明された．汚れによる耐障害性が非常に高く，非対称なので，向きの検出も可能．英数字で最大 4296 文字まで記録が可能

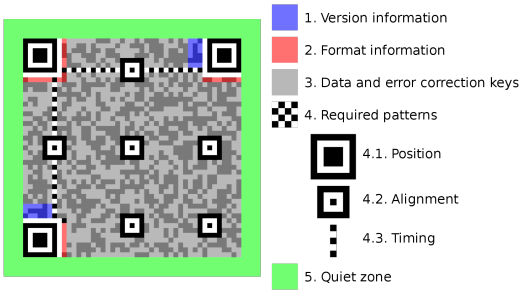

図 2.1: QR コードの仕様: https://en.wikipedia.org/wiki/QR_code

[1]世界座標系におけるカメラ座標系の座標を求める為の回転と変換のパラメータ

第3章　本研究における問題定義と仮説

本章では 1 章で述べた背景より，現状の自律航行システムを倉庫内で物品管理に用いる場合の問題点を列挙し，整理する．また，どのように解決すれば良いのかを定義する，

3.1　本研究における問題定義

現状の自律飛行システムを倉庫内での物品管理に用いる場合の問題点を列挙し，整理する．

3.2　自己位置推定の現状の問題

現状ロボットの自己位置推定に用いられる手法は大きく分けて 2 つあり，それぞれ問題がある．一つは GPS を用いた自己位置推定を行う際の利用可能環境の限定性や精度である．もう一つは LIDAR 等を用いた **??**章で説明されるような SLAM を行う際のその利用可能環境である．

3.2.1　GPS を用いた自己位置推定における問題

現在世界座標系における自己位置推定を行う上で頻繁に用いられるが，その利用可能環境には制限がある．それは屋内や遮蔽物の多い環境で衛星電波を受信出来ず利用できないという事である．加えて屋外の見通しの良い環境であっても誤差が平均して 1 2m 程あり[1]細かい制御には向かない．

一方で現在 GPS の精度向上や補助をする為のシステムが複数開発されている．その中の一つが A-GPS[1] と呼ばれ主に携帯電話やスマートフォンに搭載されているシステムである．A-GPS は GPS の信号強度が弱い場合や，建造物などの影響によるマルチパスが発生している際に利用可能なネットワークを利用して測位者の概ねの位置を取得する方式である．これにより GPS 測位時間の短縮や測位可能なエリアの拡大を実現している．

他に JAXA が開発した IMES(Indoor MEssaging System)[2] という屋内に GPS と同じ電波を利用した送信機を設置し細かい位置測位を行うという物もある．しかしこれらの方法は専用の送信機や通信機構を用意する必要がある．

[1]同研究室の水野史暁氏と竹瀬浩行氏と他数名によって慶應義塾大学湘南藤沢キャンパス WEST 地区にて計測

3.2.2 LiDAR を用いた SLAM における問題

屋内で自己位置推定を行う方法として SLAM が挙げられる．

Visual SLAM における問題

Visual SLAM はその特性上比較的多くの計算リソースを必要とする．計算機の小型化が進んでいるものの，手のひらサイズ程度の小型のロボット上で利用するには少々パワー不足となることが多い．また，この問題を避ける為計算サーバのようなものを用意する事を考えると複数機体利用する場合にネットワーク帯域や計算リソースが問題となり，解消できるような環境を用意する場合結局費用面で実用には向かない．

LiDAR SLAM における問題

LiDAR SLAM は Visual SLAM に比べて計算リソースは不要なものの，そのセンササイズがカメラよりも大きい物が多い．これは小型のロボットに積載するには向かず，ドローンのような積載重量によって航行性能に大きく影響が出るようなロボットでは尚のこと利用が難しい．役割が異なるので純粋に価格で比較することは難しいが，センサ 1 個あたりの費用も SLAM に必要なスペックのものという前提で選出するとカメラモジュールよりも高くなる．

3.3 提案手法

QR コード使わんでやるで．

3.3.1 各アプローチとの比較

ここに表を入れる．

第4章　提案手法

本章では提案手法について述べる．

4.1　概要

本研究では機材の都合上映像からの自己位置推定，機体コントロールの計算処理を遠隔PC上で行う．

全体のシステム構成は以下の図 4.1 の通りである．

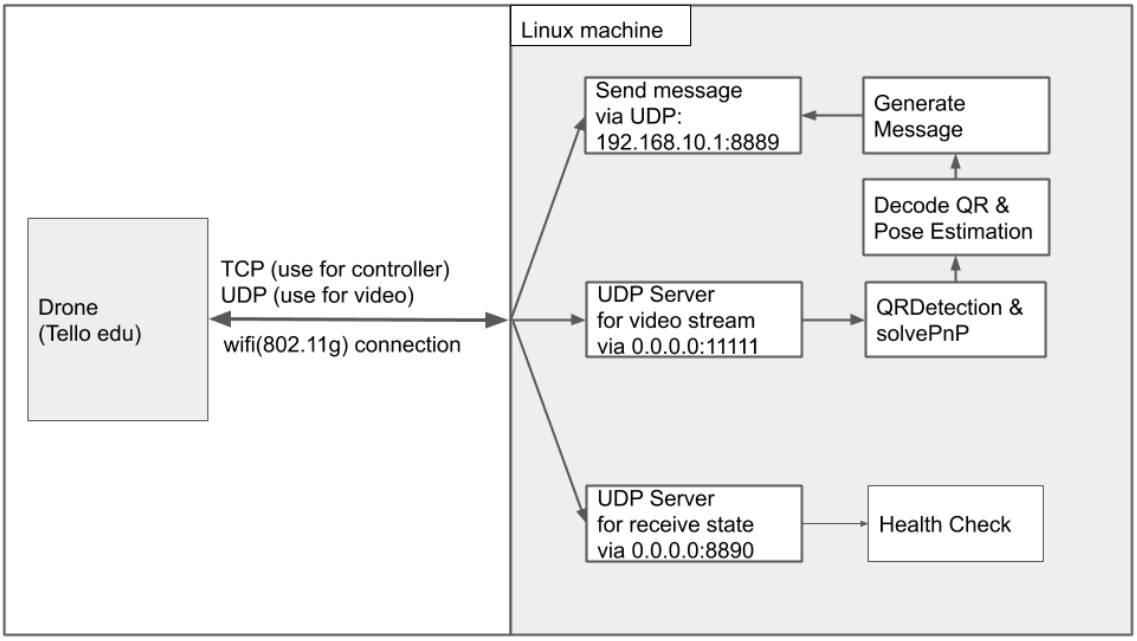

図 4.1: システム概要図

本研究ではROSの使用を前提としたシステム設計となっている．

4.2　コントローラ部

ナビゲーション部で生成された制御命令を実際にドローンに伝える部分．機体のAPIのラッパーの役割を担う．

4.3 映像部

映像を取得し，扱い易い形式に変換する部分．今回は UDP Server を建ててドローンから送られてくる映像を受け取る．

4.4 トラッカー部

QR コードの検出とデコードには OpenCV に用意されている QRCodeDetector クラスを利用し，そこから得られた QR コードの四角に対して P4P による 3 次元再構成を行った．

これにより QR コードよりデコードされた次の QR コードの位置情報とカメラ座標系からみたモデルの回転ベクトル，並進ベクトルが得られるのでこれらの情報をナビゲーション部**??**にて処理をする

4.5 ナビゲーション部

QR コードから得られたカメラの情報と次点の QR コードの位置情報を元にドローンに対してメッセージを送信する．並進ベクトルを cm 単位に変換した後そのまま移動量として利用する．また，回転ベクトルのうち y 軸 (鉛直方向) 回転のデータのみ利用する．これは今回利用している機体が基本制御として機体を水平に保つ為である．

第5章　実装

本章では提案手法の実装について述べる.

5.1　実装環境

本研究では以下の環境で実装及び実行を行う.

- DJI Tello edu
- Ubuntu 18.04
- ROS melodic
- python 3.7, 2.7

5.2　コントローラ部

プログラム 5.1: controller.py

```
1  def go_to_target(self):
2    print('target␣pos:␣', self.target_pos)
3    print('self␣pos:␣', self.drone_pos)
4    move_x_val = int(self.target_pos["x"] - self.drone_pos["x"] -
         (self.QR_SIZE/2))
5    move_y_val = int(self.target_pos["y"] - self.drone_pos["y"] -
         (self.QR_SIZE/2))
6    move_z_val = int(self.target_pos["z"] - self.drone_pos["z"])
7    rotate_val = int(self.target_pos["rotate"] - self.drone_pos["
       r"])
8    self.is_moving = True
9    if move_x_val > 0:
10       self.drone.move_right(move_x_val)
11       time.sleep(move_x_val/10 + self.COMMAND_WAIT_BUFFER)
12       print('sleep␣x:␣', move_x_val/10 + self.
             COMMAND_WAIT_BUFFER)
13   else:
14       self.drone.move_left(-move_x_val)
15       time.sleep(-move_x_val/10 + self.COMMAND_WAIT_BUFFER)
16       print('sleep␣x:␣', -move_x_val/10 + self.
             COMMAND_WAIT_BUFFER)
17
```

```
18    if move_y_val > 0:
19        self.drone.move_down(move_y_val)
20        time.sleep(move_y_val/10 + self.COMMAND_WAIT_BUFFER)
21        print('sleep␣y:␣', move_y_val/10 + self.
              COMMAND_WAIT_BUFFER)
22    else:
23        self.drone.move_up(-move_y_val)
24        time.sleep(-move_y_val/10 + self.COMMAND_WAIT_BUFFER)
25        print('sleep␣y:␣', -move_y_val/10 + self.
              COMMAND_WAIT_BUFFER)
26
27    if move_z_val > 0:
28        self.drone.move_forward(move_z_val)
29        time.sleep(move_z_val/10 + self.COMMAND_WAIT_BUFFER)
30        print('sleep␣z:␣', move_z_val/10 + self.
              COMMAND_WAIT_BUFFER)
31    else:
32        self.drone.move_backward(-move_z_val)
33        time.sleep(-move_z_val/10 + self.COMMAND_WAIT_BUFFER)
34        print('sleep␣z:␣', -move_z_val/10 + self.
              COMMAND_WAIT_BUFFER)
35
36    if rotate_val > 0:
37        self.drone.rotate_cw(rotate_val)
38    else:
39        self.drone.rotate_ccw(-rotate_val)
40    time.sleep(self.COMMAND_WAIT_BUFFER)
41    self.is_moving = False
42    self.is_standby = False
43    self.drone_pos_list = []
44    self.drone_rotate_list = []
45    return
```

5.3 映像部

今回は SDK として用意されていた libh264decoder という C++のプログラムを利用した．また，そこから得られた画像を利用し易い形に以下プログラム 5.2 を用いて利用した controller では tello SDK で用意されている tello クラスを継承して利用している．

ここでは self.drone という変数で参照が可能となっている

プログラム 5.2: controller.py

```
1  def video_loop(self):
2      try:
3          time.sleep(0.5)
4          while not self.stopEvent.is_set():
5              frame = self.drone.read()
6              if frame is None or frame.size ==0:
7                  continue
8              self.pnp_qr(frame)
9      except( KeyboardInterrupt, SystemExit):
10         self.stopEvent.set()
11         print('exit␣video␣loop')
```

5.4 トラッカー部

トラッカーノードの実装には OpenCV を用いる．

QR コードの検出とデコードには OpenCV に用意されている DetectQR クラスを利用し，そこから得られた QR コードの四角に対して PnP による 3 次元再構成を行った．PnP には OpenCV に用意されている solvePnPRansac 関数を利用した．

これにより QR コードよりデコードされた次の QR コードの位置情報と QR コードに対するカメラの回転ベクトル，並進ベクトルが得られるのでこれらの情報をナビゲーション部 5.5 にて処理をする

実際にスマートフォン上に表示した QR コードに対して姿勢推定結果を描画したものを以下の図 5.1 に示す

図 5.1: QR コードの姿勢推定結果

ソースコードの該当部分をそれぞれ以下のプログラム 5.3 に示す．本コードでは QR コードの認識時にノイズが乗る事が多く影響を減らす為に 20 フレーム分バッファを保持するようにし，その平均値を用いて制御を行なっている．

プログラム 5.3: pnp_qr.py

```
1 def pnp_qr(self, frame):
2     img = cv2.cvtColor(frame, cv2.COLOR_RGB2BGR)
3
4     qr = cv2.QRCodeDetector()
5     data, points, _ = qr.detectAndDecode(img)
6     if data:
7         _, origin_rvec, origin_tvec, inliers = cv2.
              solvePnPRansac(self.objp, points, self.mtx, self.
              dist)
```

```
8           if not self.is_moving:
9               self.target_pos = json.loads(data)
10              tvec = origin_tvec * self.UNIT_SIZE
11              rvec = origin_rvec * self.RAD_UNIT
12              self.drone_pos_list.append(tvec)
13              self.drone_rotate_list.append(rvec)
14              if len(self.drone_pos_list) > 20:
15                  self.drone_pos_list.pop(0)
16                  self.drone_rotate_list.pop(0)
17              drone_avg_pos = np.mean(self.drone_pos_list, axis
                    =(0))
18              drone_avg_rotate = np.mean(self.drone_rotate_list,
                    axis=(0))
19              self.drone_pos = {
20                  'x': drone_avg_pos[0],
21                  'y': drone_avg_pos[1],
22                  'z': drone_avg_pos[2],
23                  'r': drone_avg_rotate[1]
24              }
25
26          imgpts, jac = cv2.projectPoints(self.axis, origin_rvec,
                origin_tvec, self.mtx, self.dist)
27          img = self.draw(img, points, imgpts)
28
29          cv2.imshow('img',img)
30          cv2.waitKey(10)
31      else:
32          cv2.imshow('img',img)
33          cv2.waitKey(10)
```

5.5 ナビゲーション部

QR コードから得られたカメラの情報と次点の QR コードの位置情報を元にドローンに対してメッセージを送信する.

第6章　評価

本章では，提案システムの評価について述べる．

6.1　評価方法

倉庫内を模した周回コースを作成し，QR コードの数を変化させた際の周回成功率と速度を計測する．QR コードの数を増やせばコース上の任意の QR コードから次点の QR コードまでの距離も短くなり，カメラの計測誤差等による経路誤差も小さくなり周回成功率も向上すると考えられる．一方で本システムでは QR コードを認識するとそこで一時停止する為，QR コードの数を減らせば次点へ向かうまでの時間が短くなり周回速度が向上すると考えられる．

第7章　関連研究

基本的にドローンが自律飛行を行う際に必要とされる処理がいくつかある．そのうちの2つが自己位置推定と姿勢推定である．自己位置推定とは，ドローンが飛行している空間の中で自分がどの位置にいるのかを認識する事である．姿勢推定とは自分が水平方向に対してどの程度傾いているのか，鉛直方向に対してどの程度回転しているかといった事を認識することある．

7.1　Nanomap

これらを行なっている関連研究として Nanomap[3] という飛行モデルがある. においては 2D-LIDAR を用いた飛行経路探索手法が取られており，取得した点群データを過去数フレーム分を記憶しておき，過去の数フレーム分のデータと現在のフレームとの点群の変量によっておおよその障害物の位置を認識している．こちらの手法では，物体を正確に捉えることは放棄し，不確実な部分を物体の存在している可能性のある範囲として捉えている．視野にある物体のある可能性も含めて1番物体が少ない方向を飛行経路として選定し，飛行している．

7.2　Octomap

他に，ドローンに限らずロボット工学全般で利用されてきた Octomap[4] がある．こちらも Nanomap と同様に LIDAR を用いて周辺の環境情報を扱うものであるが，こちらは LIDAR を用いて実際に周囲を点群からモデリングして周辺状況を把握する．

Octomap ではモデリングする際に近い点群同士を同じ物体としてまとめ1つのブロックにするということを繰り返し，樹構造的にブロックを生成する為，生成後のデータは繰り返す回数等のパラメータによっては小さく圧縮することもできる. しかし，実際に点群からモデルを生成するとなるとその計算量は非常に多く，高性能な小型コンピュータが現れている現在においても処理負荷が大きく，特にドローンにおいては積載量に制限がかかりやすい為扱いにくいものとなっている．

図 7.1: 圧縮率の違いによる生成物の違い

7.3 実用性を備えた手のひらサイズ･完全オンボード処理 UAV

また，これらのように LIDAR を使用せず，CMOS センサーのみで自律飛行を行なっている例もある．

実用性を備えた手のひらサイズ・完全オンボード処理 UAV のための 3 次元自己位置推定手法の提案と全自動飛行の実現では自己位置推定で使用するセンサーは CMOS センサーを利用した自律飛行を実現している．CMOS センサーからの映像を FPGA にストリームして FAST による特徴点抽出処理を行い，その結果を Brief を用いて表現し，MCU に流し込み処理している．この研究での目的は外部処理系に依存せず，ドローン上で全て完結する自律飛行可能な小型ドローンの開発で，限定条件下 (新聞紙を敷き詰めた床面) の上で飛行させ，特徴点を十分に検出できるようにした上で，飛行させ想定通りの飛行を実現している．

7.4 DeepDrone

他に今回の研究のベース物として DeepDrone が挙げられる. こちらではドローンレースを自律飛行で行う為に自律飛行モデルを提案している. ドローンレースとは基本的にゲートを順に潜りながら飛行することが条件となっているドローン競技である. このゲートを RGB カメラで撮影した映像を推定モデルに対し,200*300 の画像を所与として $\{\vec{x}, v\}$, を推定結果として得る.$\{\vec{x}$ は $\vec{x} \in [-1, 1]^2$ と定義され正規化された入力画像の中にある目標とするゲートへの方向を表していおり, $v \in [0, 1]$ は飛行速度を正規化して表している.

DeepDrone では訓練にシミュレータでのデータと現実世界でのデータを使っている. 両者とも理想とする軌道とのズレを計算しそのズレを損失関数に入力し学習を進めていく. シミュレータ上では直接理想とする軌道とのズレを取得し, 実空間では実際に手動で実際のコース上を運び, ゲートを通過させるなどしてデータを収集していた.

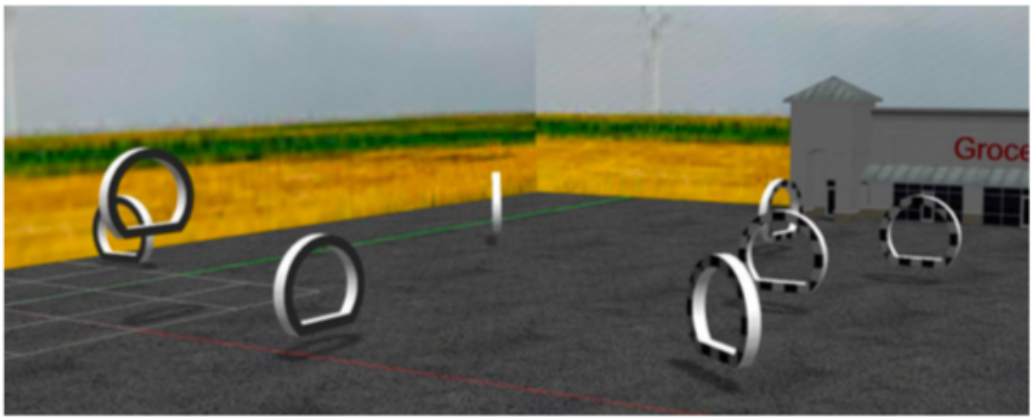

図 7.2: シミュレータ上のゲート

7.5 TDOA を用いた UAV の自己位置推定

他に屋内でドローンの位置測位を行っている例として Scalable and Precise Multi-UAV Indoor Navigation[5] が挙げられる．これは TDOA-UWB という方式を用いたものである．TDOA(Time Differences of Arrival) とは電波を用いた位置推定手法であり TOA(Time of Arrival) とよく比較される．Time of Arrival では電波を発信する送信機と位置推定を行う受信機との構成で行われる．送信機と受信機で時刻同期を行った上で，各送信機からの伝播時間を計測．この電波時間から距離を計算し，複数の送信機を用いると位置推定を行う事が出来る．

一方で TDOA では時刻同期の必要はなく，送信されてくる電波の時間差を用いて位置推定を行う．今回挙げているこの例では TDOA による位置測位を行うのに UWB を電波として用いている．

第8章　結論

本章では，本研究のまとめと今後の課題を示す．

8.1　本研究のまとめ

今回のこのシステム実際に使う上では間違って他の QR コードを認識してしまったり，悪意ある人間によって QR コードがすり替えられていないかの検証システムが必要になってくる．その場合には jwt などの方式で QR コード内のデータに署名をする．という事で対応できるが，悪意ある人間がその環境に存在した場合 QR コードの場所が置き換えられる可能性や複製されてしまうと言う可能性は残る．

また，QR コードを LED パネルで表示する方法 [6] も考案されている為この方法を用いて QR コードを容易に変更・配置の記録をする事も可能になる．一方でどこにどの QR コードが配置されているかはシステム全体として保持しておきたい情報でもあり，LED パネルで QR コードを表示する手法は本研究との親和性が非常に高いと考えられる．

8.2　本研究の課題

付 録A　付録

A.1　付録内容

謝辞

参考文献

[1] 鈴木 喬, 青木 智治, 髙橋 誠, and 緒方 進. スマートフォン向け位置測位方式の高度化. NTT DOCOMO Technical Journal, 2014.

[2] 吉冨 進. 日本発の屋内外シームレス測位の実現へ. http://www.jaxa.jp/article/special/michibiki/yoshitomi_j.html, 2003.

[3] Peter R. Florence1 et al. Nanomap: Fast, uncertainty-aware proximity queries with lazy search over local 3d data. arXiv, 2018.

[4] Hornung et al. Octomap: An efficient probabilistic 3d mapping framework based on octrees. A.Hornung and K.M.Wurm and M.Bennewitz and C.Stachniss, and W.Burgard, 2013.

[5] Tiemann Janis and Wietfeld Christian. Scalable and precise multi-uav indoor navigation using tdoa-based uwb localization. International Conference on Indoor Positioning and Indoor Navigation, 2017.

[6] Ukida Hiroyuki, Miwa Masafumi, Tanimoto Yoshio, Sano Tetsuya, and Hideki Yamamoto. Visual uav control system using led panel and on-board camera. IEEE International Instrumentation and Measurement Technology Conference (I2MTC), 2013.