Vol.2019-SLDM-189 No.6
Vol.2019-EMB52 No.6
2019/11/13

ドローンによる荷物配送経路可視化ツールの開発

疋田拓万[1]　舟橋勇佑[1]　冨山宏之[1]

概要：近年、ドローンによる荷物配送が注目されている。配送経路の探索アルゴリズム研究は数多くなされているが、可視化ツールの開発はなされていない。経路を可視化することで，顧客に荷物の位置を知らせることができる。また，配送用ドローンの位置の管理も容易となる。そこで、本研究ではドローンの消費エネルギーを最小限に荷物配送できる経路を図示するツールの作成を行った。

キーワード：配送計画問題

1. はじめに

近年，無人航空機やドローンが注目を浴びている． 中でもドローンによる荷物配送は，交通渋滞の影響を受けないほか，車両では配達の難しい地域への配達を容易に行うことが出来るため，世間の関心を集めている[1][2]． しかし，ドローンはバッテリーの駆動時間が限られており，１０分以上の飛行は困難である。そこで，よりバッテリーの消費が少ない荷物配送経路の導出が重要な問題となる． このようなドローンによる配送計画問題（VRPD）は，巡回セールスマン問題(TSP)を拡張したものとして研究がおこなわれてきた[3][4][5][6]． 多くの研究者はこの問題を解決するため，総飛行距離の最小化や総消費電力の最小化の手法などを提案している． このようにドローンによる配送計画問題を解く手法の研究は盛んにおこなわれているものの，導出した経路を直感的に理解できるように可視化するツールを開発するような研究は行われていない． より良い手法の研究を行うためには，可視化技術は重要な立ち位置であると考えられる． 導出経路の質の向上には，人間の直観的理解も重要である．

本論文では，我々が開発した可視化ツールについて述べる． 本ツールは，ドローンによる配送経路のうち最もエネルギー消費の小さい経路を出力する． 探索する経路の定義は，配送基地から出発し，各配送地点を 1 度ずつ通過しながら荷物を配送し，すべての荷物の配送後，配送基地に戻るまでの経路を探索することとする． それらの経路のうち，最もエネルギー消費の小さい経路を導出する． このような問題は最小消費エネルギーを求める配送計画問題(EMVRP)と呼ばれ，[7]に基づいている．開発した本ツールでは全探索法（ＢＦ），動的計画法（ＤＰ），最近傍探索（ＮＮ）の３つの解法で経路を探索する．

本論文の構成は以下のとおりである． 2 章では，関連研究について述べる． 3 章では，今回対象とする配送計画問題について述べる． 4 章では，本ツールの実装方法について述べる． 5 章では，本ツールの使用方法を述べる． 6 章では，本ツールの動作確認を行う． 7 章では，本論文の結論を述べる．

1, 立命館大学

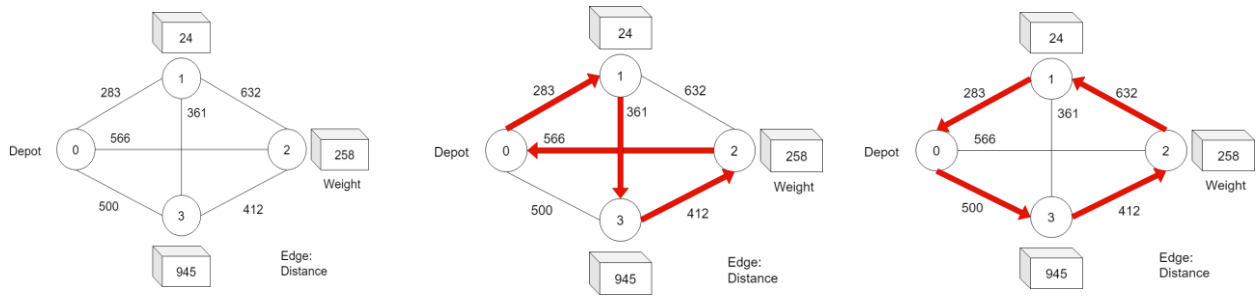
2. 関連研究

巡回セールスマン問題（TSP）は古典的な問題である． すべての都市を巡回し最初の地点に帰るような経路のうち，最も距離が短いものを求める問題である．TSP は NP 困難であることが知られており，多くの場合近似的な解法が用いられる．単純な近似解法として，最近傍探索（NN）が挙げられる．これは未配送地点の中で最も近い点を次々選んでいく形で探索を行う．厳密解を求めるアルゴリズムは，Bellman[8]や Held[9]によって開発されている．

配送計画問題（VRP）は，TSP を拡張した問題である．荷物はトラックのような車両により配送されるものとする．VRP には，様々な制約条件が存在する．制約条件の例として，配送車両が持てる荷物の上限などが存在する．

最小消費エネルギーを求める配送計画問題（EMVRP）は，エネルギー消費を考慮した VRP である．ある地点から他のある地点に移動するための消費エネルギーを数式で定義する．この数式は，二点間距離，配送車両の重量，配送物重量により構成されている．EMVRP は，全ての顧客に荷物を届けて巡回する経路のうち，最も消費エネルギーの小さい経路を求める問題である．一般的に，この問題は NP 困難とされる．文献[7]では，Kara が整数計画問題として EMVRP の定式化を行っている．消費エネルギーは距離と重量の積に比例すると仮定されている．文献[10]では，Negoro が EMVRP を解くための重さ優先アルゴリズムを提案した．これは最も重い荷物を優先的に配送するアルゴリズムである．しかし重さ優先アルゴリズムが従来の NN アルゴリズムよりも悪いことが実験から明らかになっている．Dorling[11]は、マルチモータードローンのエネルギー消費に関する詳細なモデルを開発し、焼きなまし法で EMVRP を解くアルゴリズムを提案した．Wang[12]は、ヘテロジニアスな車両に対応できるよう EMVRP を拡張し、整数プログラミングの定式化を提示した．また文献[13]では，Wang はヘテロジニアスな車両に対応した EMVRP の遺伝的アルゴリズム（GA）を提案した．Funabashi[14]は，EMVRP を動的計画法（DP）で解く手法を提案した．

Vol.2019-SLDM-189 No.6
Vol.2019-EMB52 No.6
2019/11/13



(a) 例題　(b) TSP での最適経路　(c) EMVRP での最適経路

図 1　ドローンの配送経路の例

一方で，可視化ツールの開発は広い分野において進められてきた．Daniel[15]はプログラミングの知識を持たない学生でも親しめるような，フローチャートベースのマルチエージェントシステムを開発した．Roberts[16]は，ユーザが選択したアルゴリズムの視覚化を行うツールを開発した．TSP の可視化ツールは既に開発されている[17]が，これらのツールで EMVRP を解こうとすると，ツールのインストールや，複雑な設定を行う必要がある．我々が開発したツールでは，EMVRP を解く際にインストールの必要も複雑な設定も必要としない．

3. ドローンの荷物配送における経路探索問題

この章では，本論文における EMVRP，NN，DP，BF について定義する．

3.1 EMVRP

配送物の個数を N とする．2 つの配送物を同じ顧客に配送することはできない．複数の配送物を同じ顧客に届ける場合は，事前に 1 つの配送物としてパッケージされているものとする．したがって、顧客の数も N となる．

図 1(a)に示された例を考える．「0」というラベルが付いたノードは配送基地を示し、他の 3 つのノードは顧客（配送先）を示す．箱の中の数字は配達する荷物の重量を表し、枝の数字は 2 顧客間の距離を表している．TSP における最適経路が図 1(b)に示されている．この経路での消費エネルギーは 1622 (= 283 + 361 + 412 + 566)である．しかし，この経路は EMVRP における最適解ではない．ドローンの消費エネルギーは，飛行距離だけでなく重量にも関係しているため, 今回の例では重い荷物は出来る限り早く配送すると，エネルギー消費を小さくできる．したがって，この例での消費エネルギーが最小となる経路は図 1(c)のようになる．総飛行距離は 1827 (= 500 + 412 + 632 + 283)であり，TSP における最適経路よりも長くなっている．このように，最短経路が最もエネルギー効率の良い経路とは限らない．

本論文では、全ての荷物が 1 回の飛行で配達されることを想定している．全ての荷物が配送基地にある状態で、ドローンは配送基地から配送を開始する．ドローンが持てる荷物の上限については，今回の論文では考慮しないこととする．

ドローンは$city_1$，$city_2$，…，$city_N$に配送するものとする．$city_i$から$city_j$に配送する際ドローンが持つ荷物の総重量をW_{ij}とする．また，$city_i$から$city_j$の距離をd_{ij}とする．$city_i$から$city_j$での消費エネルギーE_{ij}は式(1)で定義される．

$$E_{ij} = 0.04 * (300 + W_{ij}) * d_{ij} \quad (1)$$

N!通りの配送経路のうち，E_{ij}の合計が最も少ない経路を探索する．

3.2 最近傍探索法

最近傍探索法（NN）は，計算幾何学で盛んに研究されている近似手法の 1 つである[18][19]．NN による探索は巨大な問題でも高速に計算結果を導出できるが，NN は厳密な最適経路を導出しない場合もある．本研究における NN は，次に配送する顧客を決める際，最小の消費エネルギーで配送できる顧客を選ぶという手法である．

3.3 動的計画法

動的計画法（DP）は，一般的に数学的最適化問題へのアプローチである．DP では，特定の問題を再帰的な手法で部分問題に分割する．小問題の最適解を用いることで，元の問題の最適解を導出することができる．本ツールでは. [14]のアルゴリズムを用いている．

3.4 全探索法

全探索法は，単純な解決策である．解決策として考えられるすべての候補を体系的に列挙し，各候補が問題の条件を満たしているかどうかを確認するという手法である．本論文では，すべてのルートのエネルギー消費量を計算し、候補のうち最適な経路を選択する．N が 8 より大きくなる場合には，組み合わせ爆発によりこの手法で最適経路を求めることはできない．

4. 開発環境と仕様

この EMVRP 可視化ツールの開発において，flask[20]を使用した．flask は Web アプリケーションを開発するフレームワークである．バージョンは 1．0．2 である．計算部分には python3．7，GUI の構築部分には HTML と JavaScript を用いた．この EMVRP 可視化ツールは特定の URL で閲覧することが出来る．ユーザは Web 上で，配送地点の座標など

Vol.2019-SLDM-189 No.6
Vol.2019-EMB52 No.6
2019/11/13

必要な情報が書かれた Excel ファイルをアップロードすることで，NN，DP，BF の 3 手法のうちいずれかを選びその結果が図示されたものを見ることが出来る．

5. 実装

この章では，本ツールの実装方法について述べる．本ツールの開発には，flask[20]を使用した．flask は Web アプリケーションを開発するフレームワークであり，単純なアプリケーションの開発に適している．flask で製作するアプリケーションは主に Python で記述されたファイルや，JavaScript や CSS を含んだ HTML ファイルで構成されている．メインの Python ファイルが立ち上がると HTML ファイルを呼び出し，ページ遷移を行う．ユーザがファイルのアップロードを行うと，Python ファイルが NN，DP，BF のいずれか指定された手法で EMVRP を解く．HTML ファイルはその結果を受け取り，これに基づいて図の生成を行う．この動きの流れの概念図は図 2 のようになる．

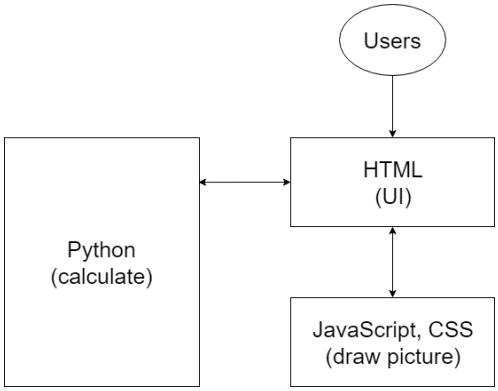


図 2 概念図

フローチャートは図 3 のようになる．最初に，ユーザは HTML にデータを入力する．このデータには配送都市数や都市の座標，荷物の重量が含まれる．ユーザが submit ボタンをクリックすると，これらのデータは Python ファイルに渡される．Python ファイルはこれらのデータを用いて NN, DP, BF の計算を実行する. Python ファイルは導出した経路の情報をブラウザで可視化する．図の表示の際には，Python ファイル内で正規化処理を行い，ウィンドウサイズが過剰に大きくならないよう設計されている．

$city_i$の x 座標をx_iとする．同様に，$city_i$の y 座標をy_iとする．x_iの最大値をx_{max}，y_iの最大値をy_{max}とする．x_i，y_iを正規化したものは$scalepoint_x$，$scalepoint_y$とする．本ツールは図の幅を 50px から 650px の間に正規化するため，正規化のための関数は式（2）および（3）のようになる．

$$scalepoint_x = (700 - 100) * \frac{(x - x_{min})}{x_{max} - x_{min}} + 50 \qquad (2)$$

$$scalepoint_y = (700 - 100) * \frac{(y - y_{min})}{y_{max} - y_{min}} + 50 \qquad (3)$$

Python ファイルは，導出経路，$scalepoint_x$，$scalepoint_y$を HTML ファイルに送る. HTML ファイルは図を生成するため，$scalepoint_x$，$scalepoint_y$を JavaScript に送る．

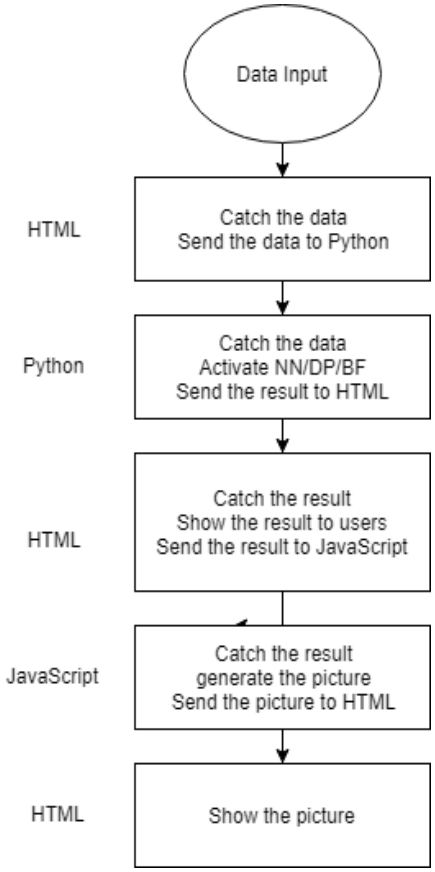


図 3　フローチャート

図の生成には，JavaScript の機能である canvas を使用する. JavaScript の canvas で定義された形式で描画する図を指定することにより，長方形や円，その変形，色やグラデーション，画像やテキストとの合成などの表現を含むさまざまな図形の描画が可能になる. canvas を用いると, 単純な方法で図を生成することが出来る．canvas には矢印を簡潔に生成する方法が無いため，canvas-arrow というプラグインを使用した．始点と終点の情報があれば，簡単に矢印を描画できる．

最後に，HTML ファイルが図のデータを受け取る．ユーザが図の生成ボタンをクリックすると，HTML は図を表示する．

本ツールは Web ページを装飾するのに CSS ファイルを使用している．CSS は文字や背景のデザインに特化した言語である．

また本ツールは共有のために heroku[21]を用いているた

Vol.2019-SLDM-189 No.6
Vol.2019-EMB52 No.6
2019/11/13

め，どのパソコンでも本ツールにアクセス可能である．

6. 動作確認

この章では，本ツールの動作確認を行う．最初に，ユーザは配送する都市の座標とそれぞれに配送する荷物の重量が記述された Excel ファイルを用意する必要がある．記述方式は図 4 のようになる．A 列が各都市の x 座標，B 列が各都市の y 座標，D 列が各都市に配送する荷物の重量を示している．

x-axis　y-axis　weights of items

	A	B	C	D
1	0	0		0
2	1	1		3
3	2	4		2
4	1	5		1
5	-2	3		7
6	3	1		4
7	5	9		2
8				

図 4　Excel ファイルの記述例

初期画面は図 5 のようになっている．ユーザは，NN, DP, BF のうちどの手法で解くのか選択することが出来る．例えば NN を選択したい場合は，図 6 のように入力し，先ほど用意したファイルをアップロードする．

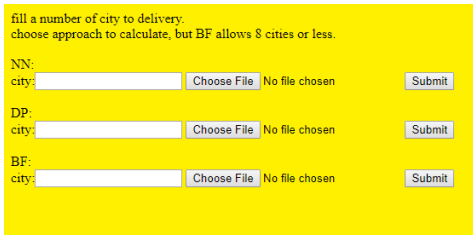


図 5　初期画面

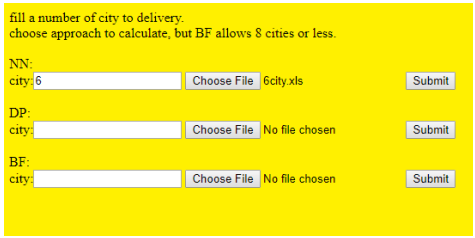


図 6　実行直前の画面

ユーザが submit ボタンをクリックすると，EMVRP の結果を表示するページに遷移する．generate ボタンをクリックすると，図 7 のように目的の図が表示される．他の手法で生成される図を見る場合は，同様の手順を踏むことで続けて実行することが出来る．

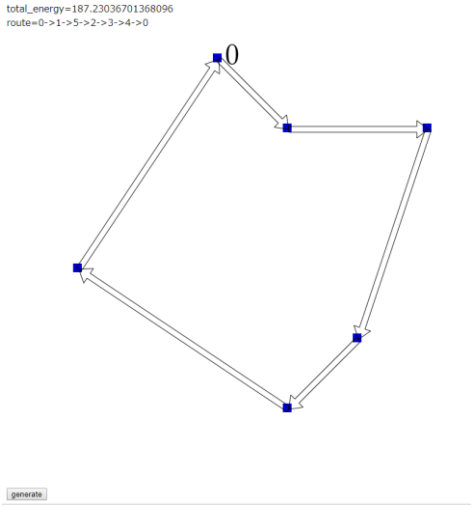
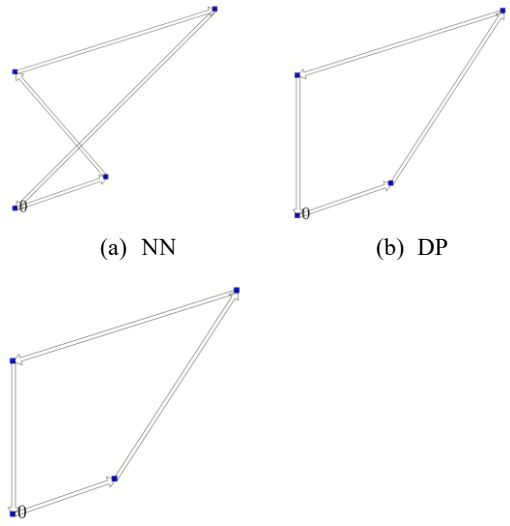


図 7　出力図

7. 妥当性の確認

この章では，妥当性の確認としてテストを行う．本ツールで導出される総消費エネルギーは，複数のテストケースの結果を比較する必要がある．総消費エネルギーの数値の正確性を確かめるため，Funabashi[14]のツールを用いた．配送地点までの距離を計算し，それを入力として使用することにより，同じ結果が得られるかどうかを確認した．数値や経路について，全く同様の結果を得ることができた．

本ツールの NN では，必ずしも最適経路を導くとは限らない．目前に早く配送できる地点があれば，それが原因で最適経路を導出できない場合がある．このような例は，図 8 で示されている．



(a) NN　(b) DP

(c) BF

図 8　NN が違う経路を出力する例

この例では，配送地点の座標は (-7, 10) , (4, -9) , (-2, 7) ,

Vol.2019-SLDM-189 No.6
Vol.2019-EMB52 No.6
2019/11/13

（-7, -3）である．最初の座標は配送基地で，残りの３つの座標が配送先の座標である．配送する重量はそれぞれ 4, 9, 9 である．DP と BF は（-7, -3）を最後に通る経路が最適という結果を出したが，NN は（-7, 10）を最後に通る経路を導出した．（4, -9）は配送基地である（-7, 10）から十分に遠い為，次の配送先として（4, -9）を最後まで選ばなかったと考えられる．

続いて，計算できる顧客数の限界を確認する実験を行った．NN の場合では，生成した図は見づらくなったが，顧客数を 1000 にした場合でもツールは停止せず瞬時に結果を導出した．DP の場合での顧客数の限界はある条件下で 16 であった．ある条件下とは，各座標の x 座標と y 座標の値がともに-10 から 10 以内の整数であり，各顧客に配送する荷物の重量を 0 から 10 の間の整数とした場合である．それ以上の顧客数で計算を行おうとすると，本ツールはエラー画面を表示する．BF の場合では，顧客数 9 が限界であった．

図 9 は，3 つのアルゴリズムの実行時間と顧客数をグラフ化したものである．10 個のサンプルデータを Excel で生成し，それらのデータを入力した際の計算時間の平均をとったものが縦軸の実行時間である．横軸は顧客数である．

NN と DP は，それぞれ顧客数 7，10 から指数関数的に実行時間が増えることが分かった．

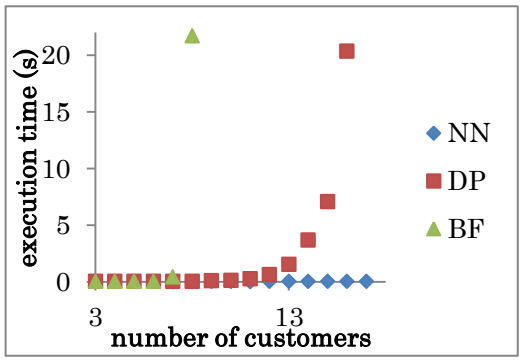


図 9　実行時間

図 10 は，3 つのアルゴリズムで導出した経路の消費エネルギーの値の差を DP で正規化したグラフである．NN は，DP や BF と比べて消費エネルギーが大きい経路を求める傾向にあることが分かった．

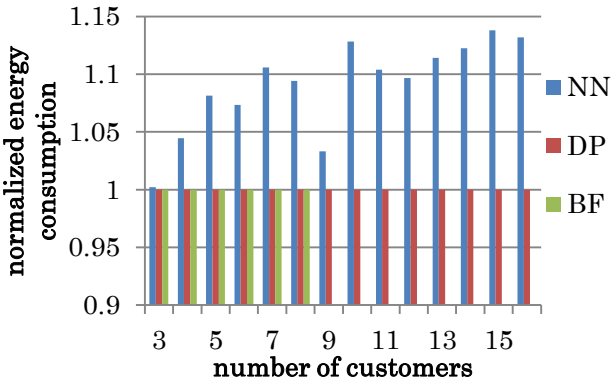


図 10　消費エネルギーを正規化したグラフ

8. おわりに

近年ドローンは自宅への配送サービスへの使用を検討されている．本研究では，ドローンの配送経路を可視化するツールの開発を行った．このツールは EMVRP の結果を 3 つのアルゴリズムで導出することができ，導出経路を図示するものである．

本ツールは配送経路の直観的理解や，配送状況の確認などに有益だと考えられる．

参考文献

[1] Amazon Prime Air.　http://www. amazon. com/primeair
[2] Project Wing.　https://x. company/projects/wing/
[3] S.　Anily, G, Mosheiov, "The traveling salesman problem with delivery and backhauls," Oper Res Lett vol.　16, pp.　11–18, 1994.
[4] S.　Anily, J.　Bramel, "Approximation algorithms for the capacitated traveling salesman problem with pickups and deliveries," Nav Res Logist, vol.　46, pp.　654–670, 1999.
[5] M.　Gendreau, G.　Laporte, D.　Vigo, "Heuristics for the traveling salesman problem with pickup and delivery," Comput Oper Res, vol.　26, pp.　699–714, 1999.
[6] R.　Baldacci, A.　Hadjiconstantinou, A.　Mingozzi, "An exact algorithm for the traveling salesman problem with deliveries and collections," Networks, vol.　42, pp.　26–41, 2003.
[7] I.　Kara, B.　Y.　Kara, M.　K.　Yetis, “Energy minimizing vehicle routing problem,” International Conference on Combinatorial Optimization and Applications, Springer, pp 62-71, 2007.
[8] R.　Bellman, “Dynamic programming treatment of the travelling salesman problem,” Journal of the ACM, vol 9, pp.　61-63, 1962.
[9] M.　Held, R.　M.　Karp, “A dynamic programming approach to sequencing problems,” Journal of the Society for Industrial and Applied Mathematics, vol 10, pp.　196-210, 1962.
[10] S.　Negoro, I.　Taniguchi, H.　Tomiyama, “Fundamental analysis of low energy path routing for delivery quadcopters,” International Technical Conference on Circuits/Systems, Computers and Communications, 2016.
[11] K.　Dorling, J.　Heinrichs, G.　G.　Messier, S.　Magierowski, “Vehicle routing problems for drone delivery,” IEEE Transactions on Systems, Man, and Cybernetics: Systems 47:70-85, 2017.
[12] S.　Wang, X.　Liu, “Energy minimization vehicle routing problem with heterogeneous vehicles,” International Conference on Service Systems and Service Management, 2016.

[13] S. Wang, Y. Wu, “A genetic algorithm for energy minimization vehicle routing problem” International Conferemce on Service Systems and Service Management, 2017.
[14] Y. Funabashi, A. Shibata, S. Negoro, I. Taniguchi, H. Tomiyama, “A dynamic programming algorithm for energy-aware routing of delivery drones,” Artificial Intelligence and Data Engineering, 2019.
[15] H. Danial, B. A. Rodina, G. R. Ram, H. N. M. N. Mohd, Y. Moslem, H. Shi-Jinn, R. Jože, “A flowchart-based multi-agent system for assisting novice programmers with problem solving activities,” Malaysian Journal of Computer Science, Vol. 28(2), pp. 132-151, 2015.
[16] J. C. Roberts, P. D. Ritsos, J. R. Jackson. , C. Headleand, “The explanatory visualization framework: An active learning framework for teaching creative computing using explanatory visualizations,” IEEE transactions on Visualization and Computer Graphics vol 24, no. 1, pp. 791–801, 2018.
[17] Google optimization tool. https://developers. google. com/optimization/
[18] L. B. Jon, W. W. Bruce, C. Y. Andrew, "Optimal expected-time algorithms for closest-point problems," ACM Tran. Math. Software, vol 6, pp. 563-579, 1982.
[19] L. C. Kenneth, "Fast algorithms for the all nearest neighbors problem," 24th Annual Symposium on Foundations of Computer Science, pp. 226-232 1983.
[20] Flask. https://a2c. bitbucket. io/flask/
[21] Heroku. https://jp. heroku. com/