The 11th NAPROCK International Programming Contest, Hanoi, Vietnam

Book of Abstracts

Competition Section

Carrot / NIT, Tokyo College, Japan

Agents who do not dance are demotion / NIT, Hachinohe College, Japan

Like slugs which has become hydrangeas / NIT, Kurume College, Japan

Ultimate Magicians / Salesian Polytechnic, Japan

KUMA SYSTEM / NIT, Kumamoto College, Japan

com-PASS / NIT, Yuge College, Japan

SSBoys / Univ. of Science and Technology, VNU, Vietnam

BKDN.AlphaGo / Univ. of Science and Technology, The Univ. of Danang, Vietnam

Team: HaUi.TodayIFeelSoGood / Ha Noi Univ. of Industry, Vietnam

BKAC.Milos / Ho Chi Minh Univ. of Technology, Vietnam

HCMUE_02 / Ho Chi Minh City Univ. of Education, Vietnam

Server TimeOut / Ha Noi Open Univ., Vietnam

Revolution Technology / Univ. of Engineering and Technology, VNU, Vietnam

SIU Talent / The Saigon International Univ., Vietnam

ChromeOS / Univ. of Engineering and Technology, VNU, Vietnam

Plan Y / Univ. of Engineering and Technology, VNU, Vietnam

Fibo / National Univ. of Mongolia, Mongolia

VTC Engineering / VTC Pro-Act Training and Development Centre, Hong Kong

Carrot

Hiroki Shibata, Shota Yamaguchi, Wataru Okada, Raito Matsuzaki
National Institute of Technology, Tokyo College, Japan

1. Introduction

We change tactics in the first and second matches. In the first match, we adopt a neural network-based algorithm called ResNet. In the second match, we adopt search algorithm based on beam search. These algorithms may be changed flexibly by observing the state of the first match field and the game development of other teams. The tactics outline is described in the following chapters.

2. Tactics in the first match

In the first match, the competition field is fixed. We use a neural network to determine actions of the agents.

2.1 Network Architecture

We use an actor-critic network to select actions. Our network takes field data as input, and outputs movement policy and value of the current state. Inputs of the network contains team score, tile data, number of turns, and positions of the agents. Output policy represents the probability that each plan of actions is optimal idea. We use the residual convolutional network called "ResNet".

2.2 Supervised Learning

Before the reinforcement learning, we trained the model with supervised learning. We made self-play data using greedy algorithm. To increase the diversity of play data, we added a small noise to evaluate values. The network is supervised by self-play data for 100 epochs. In each epoch, approximately 100,000 self-play data are used.

2.3 Reinforcement Learning

After the supervised learning, we trained the network using reinforcement learning using A2C architecture. To improve search performance, we use Monte Carlo Tree Search (MCTS) in playout. As a reward function, we use the difference of the score from previous turn.

3. Tactics in the second match

In the second match, the fields are not published in advance. Therefore, we use beam search algorithm and another evaluation function instead of neural network.

3.2 Beam Search

We calculate the evaluation value of all possible actions for each agent using beam search. To reduce the calculation time, the evaluation function does not use a region data.

3.3 Action Assignment

If each agent determines the moves to maximize own evaluation value, agent actions may cause some conflicts. Agent actions should not be conflicted, and we want to maximize the total evaluation values of all selected actions. To maximize the total evaluation values, we use Hungarian Algorithm.

4. Visualizer

We made a visualizer to support the operation. The visualizer shows a field data and the score of each team. Player can change actions of agents by clicking a tile on the visualizer.

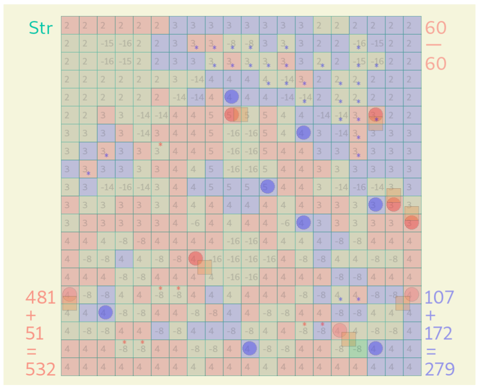


Fig.1 Screen shot of GUI.

5. System

We mainly use C++ for implementation of the tactical decision algorithm. However, in some cases (e.g. using neural network, developing the test server), we also use Python3 for the implementation since there are several libraries that are useful for the implementation.

Agents who do not dance are demotion

Ogura Naoya, Ito Rui, Kikuchi Daiki, Hosokawa Yasushi
National Institute of Technology, Hachinohe College

1. System overview

First, information necessary for participation in each game is confirmed from the game advance information acquisition API, and parameters are input using a GUI.

Next, by pressing the start button arranged on the GUI, it connects to the server according to the entered parameters and acquires information.

After that, by pressing the reload button arranged on the GUI every turn, information reception, calculation of the next traveling direction, and transmission of the calculation result are performed simultaneously.

2. About calculation algorithm

The algorithm used as the main axis is based on the minimax method.

An algorithm with increased versatility is used for a secret field, and an algorithm dedicated to each field is used for a public field. The dedicated algorithm is one in which the evaluation value is adjusted so that a high score can be obtained in the field.

The algorithm uses many ingenuities, such as making sure that allied agents are not too close and comparing the scores of each other when specifying the same area as the enemy agents and changing the behavior.

3. About GUI

This competition has a limited time, so players are required to perform simple operations. To assist development, it is necessary to grasp the situation intuitively. Therefore, these problems are solved by using GUI.

Fig.1 GUI during a match

When participating in the match, enter the match ID and team ID and press the start button. Then, the user accesses the URL corresponding to the match ID, acquires the first board, and displays the board in another window.

During the match, press the reload button once per turn. Then, the current information is obtained from the server, the best calculation is performed by the algorithm, and the calculation result is automatically transmitted to the server.

Like slugs which has become hydrangeas

Yosuke Higuchi, Yudai Inada, Yuta Saeki, Ryo Tanaka
National Institute of Technology, Kurume College
/ Department of Control and Information Systems Engineering

Introduction

In the competition, each agents' action can cause conflict. It is not a good idea to have agents act independently. However it is too huge to enumerate combinations of all actions, we enumerate combinations of some actions using some ideas and choose actions among them.

Algorithms

First, except in cases shown below, we consider opponent's agents do not exist. About our agents, instead of ignoring combinations of all our agents' action, we have agents act without getting close with each other simply.
Second, we ignore tiles under a certain standard except in some cases. This is based on experiences that we can fight without tiles because we can remove opponent's tiles and opponent can remove our tiles.
Third, agents' algorithm will change to greedier algorithm when an opponent's agent comes nearby. This is because conflicts are loss when our agents take advantageous algorithm in the long run.
Fourth, we added a program to enable our agents' action to be visible and one to enable low score tiles to be invisible. (Fig.1) We also added a program to have agents take action instructed by us to deal with our and opponent's agents' bad action that is not considered by GUI. (Fig.2)
Finally, we added algorithm used in cases that it is better to take particular action than to use usual algorithm. For example, when we can predict that an opponent's agent will move and a tile under the agent can be high score, our agents will remove the tile. (Fig.3)

Conclusions

We fight by algorithm as the above in the competition.

Development Environment

OS: Windows10, MacOS X
Language: C++, Pythonas
Library: OpenSiv3D, picojson
IDE, Editor: Xcode, Visual Studio, vim

Figure

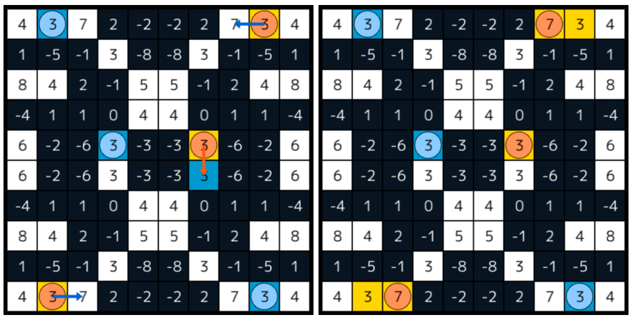


Fig.1
The left figure is before the actions and the right one is after the actions. Blue arrow is for move and orange one is for remove.

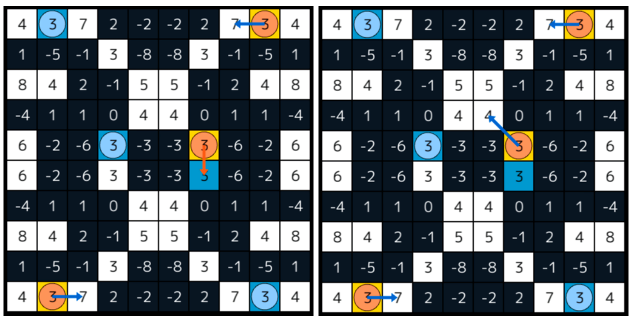


Fig.2
The action of our agent of middle of the left figure be change by us. Then, the result is shown in the right figure. We can do this instruction at any time.

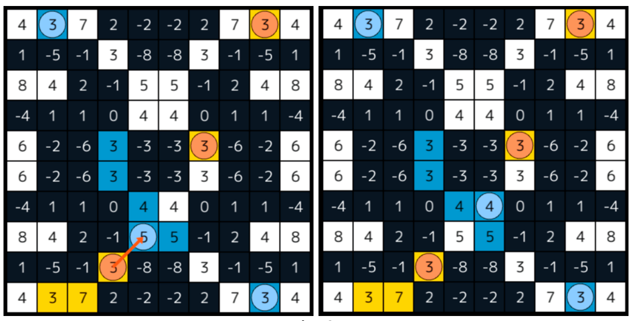


Fig.3
We can predict opponent's agent of middle of the left figure be move to upper right. So, our agent removes a tile under the agent. Then the result is shown in the right figure.

Ultimate Magicians

Yusuke Kosaka, Yuta Katsuki, Minoru Kojima, Shuichi Utsugi
Salesian Polytechnic, Japan

1. Introduction

Systems using the machine learning and complex search algorithms may be strong, however, much longer execution time in processing is required. If the execution time is too long, in the worst case, the agent's movement cannot be instructed in a given time, and they will not do anything at that turn. If this situation occurs again, player cannot win the game. Therefore, we do not select difficult method and consider a simple search algorithm which can realize a short execution time. Our system consists of the following three parts: "search", "GUI", and "communication".

2. System

2.1 Search Algorithm

For every turn, each agent must decide some movement for any of eight tiles surrounding oneself. Here stagnation is not selected as one of agent’s movement. The criteria of movement selection is the points of the eight tiles surrounding the agent, the existence of tiles, the distance between friendly agents, the point distribution of the whole field, the ability to produce a surrounded area. An evaluation value is calculated based on the above criterion, and the best movement is determined. The agent's movement is determined not by considering the combinations of the movement of agents, but by selecting the best movement of the agents one by one in order. By the above method, it is possible to determine agent's movements in a short time even in a game with many numbers of agents. It is also possible to predict several movements of few steps ahead as needed. By adopting this method, it possible to pass through a tile with negative point once and remove its tile by oneself, and hence, agents can move on the field in all directions.

2.2 Disclosed Fields

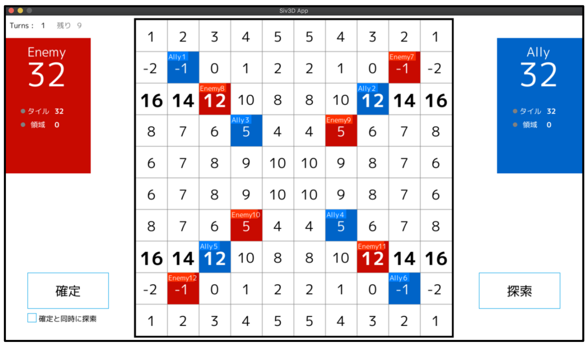
Our system analyzes the disclosed filed in advance. In the advance analysis, our system analyzes distributions of points and adjusts evaluation functions that agents easy to approach to areas with high points, and conversely, they do not approach to areas with many negative points. In particular, we define many kinds of evaluation functions and evaluate functions by repeating matches between computers. An evaluation function using in the competition is the strongest function obtained by our evaluations.

2.3 Undisclosed Fields

In games using the undisclosed fields, our system analyzes field similar to the case of the disclosed fields. First, the point distribution is analyzed at beginning the game. Second, values of each tile is obtained and our search system decides agent’s movements by using its information in our search system. The value of a square is obtained from the point of tiles, the point of adjacent tiles, and the position of tiles.

2.4 GUI

By using the GUI (Graphical User Interface), players can check the current status of the game field in real time. In our GUI, a game status is automatically updated after receiving a current game status. Information displayed on our GUI is not only the point of each tile (squared region), the configuration of tiles, and the position of agents, but also the tile point for each team, the area points, the total point for each team, and the search results by our system. In the case that the search fails, players can issue manually an instruction to an agent on our GUI. Therefore, we developed GUI that instructions for agents can easily input by players.



1	2	3	4	5	5	4	3	2	1
-2	Ally1 -1	0	1	2	2	1	0	Enemy7 -1	-2
16	14	Enemy8 12	10	8	8	10	Ally2 12	14	16
8	7	6	Ally3 5	4	4	Enemy9 5	6	7	8
6	7	8	9	10	10	9	8	7	6
6	7	8	9	10	10	9	8	7	6
8	7	6	Enemy10 5	4	4	Ally4 5	6	7	8
16	14	Ally5 12	10	8	8	10	Enemy11 12	14	16
-2	Enemy12 -1	0	1	2	2	1	0	Ally6 -1	-2
1	2	3	4	5	5	4	3	2	1

Fig1. An example of screen displayed by our GUI

2.5 Communication

In our system, after a current game information is received from the server, agent’s movements calculated by our system are transmitted to the server. In particular, players use a GUI in order to send and receive game information.

2.6 Development Environment

Our system was developed by using the programing languages C++ and Python in computers with mac OS.

KUMA SYSTEM

Suenaga Kazuhiro, Shirakawa Takahiro, Ogitsuka Kazuki, Ningping Sun
National Institute of Technology, Kumamoto College, Japan

1. Introduction

Rule of the competition is that there is a field that assigns arbitrary points to each tile, and the agents move on the field to compete for score. There are two conditions for earning the points. The first is to obtain the tile points by moving the ally agent to make the field tile an ally tile, and the second is to get the area point by enclosing the ally tile.

2. Approaches

2.1 Solver for public fields

For the public fields, using pre-obtained field information, learning is performed through machine learning using a deep Q-Network such as Alpha Zero. We use Monte Carlo tree for searching, meanwhile we need to adjust the parameters so that the inference can be completed in one turn limit. However, there is only a result of the previous competition. Comparing the result of the previous competition with the one that is adjusted by the evaluation function of the solver for private fields described later specifically for the target field, we choose the better one.

2.2 Solver for private fields

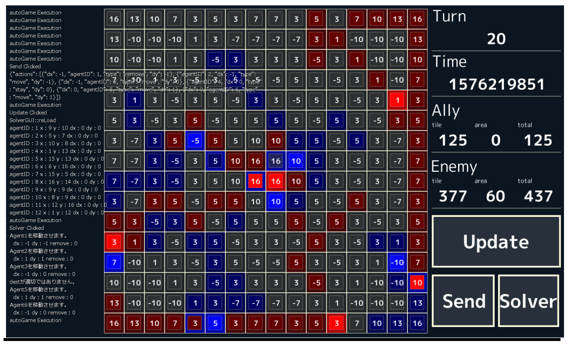
In the case of the private fields, it is important to handle any field as the general purpose. Here we need to adjust the evaluation function, judge for each situation using the beam-search, and consider the next move by adopting the result with the highest evaluation. However, because the maximum number of agents is eight, a combinatorial explosion will occur if we search as it is. By pruning nodes with a low evaluation and limiting the search depth, the next movement will be adjusted within the time limit of one turn.

2.3 Demo Competition Server

We have one more important thing additionally need to do for tuning the solver to best condition before the actual competition. It is to send the action information exactly to the contest server in the competition. It makes no sense that even if we could get the best answer on our own computer, we didn't send our exact answer to the contest server. Therefore, by preparing a virtual competition server based on the actual connection environment to simulate the competition, we can repeat the exercise until perfect. In addition, it also plays a debugging role. We use it to detect and fix errors in our solver program.

2.4 Operation GUI

We assumed that solver takes a valid action within the limited time, and also assumed that risk of taking an unexpected action such as the case that solver keeps taking action 'stay' or the case that agents take action back and force between our team tiles. An operable GUI for limiting these actions is necessary and essential that can maximize cooperation with "General-Purpose Processing Unit †NINGEN†". As shown in Fig.1 the GUI has functions including updating the field information, showing the selected action information on the field view, and submitting action information. In addition to these functions, if the solver failed to action selection, we can change the action by flicking on the display as rescuing, which makes it possible to instantly reflect the decision made by †NINGEN†.



16	13	10	7	3	5	3	7	7	3	5	3	7	10	13	16
13	-10	-10	-10	1	3	-7	-7	-7	-7	3	1	-10	-10	-10	13
10	-10	-10	1	3	-5	3	3	3	3	-5	3	1	-10	-10	10
7	-10	1	3	-5	3	5	-5	-5	5	3	-5	3	1	-10	7
3	1	3	-5	3	5	-5	3	3	-5	5	3	-5	3	1	3
5	3	-5	3	5	-5	5	5	5	5	-5	5	3	-5	3	5
3	-7	3	5	-5	5	5	10	10	5	5	-5	5	3	-7	3
7	-7	3	-5	3	5	10	16	16	10	5	3	-5	3	-7	7
7	-7	3	-5	3	5	10	16	16	10	5	3	-5	3	-7	7
3	-7	3	5	-5	5	5	10	10	5	5	-5	5	3	-7	3
5	3	-5	3	5	-5	5	5	5	5	-5	5	3	-5	3	5
3	1	3	-5	3	5	-5	3	3	-5	5	3	-5	3	1	3
7	-10	1	3	-5	3	5	-5	-5	5	3	-5	3	1	-10	7
10	-10	-10	1	3	-5	3	3	3	3	-5	3	1	-10	-10	10
13	-10	-10	-10	1	3	-7	-7	-7	-7	3	1	-10	-10	-10	13
16	13	10	7	3	5	3	7	7	3	5	3	7	10	13	16

Fig.1 Operating GUI.

3. Conclusion

Tuning parameter of the solver plays an important role in supporting †NINGEN† because †NINGEN† makes the final decision, so the operation of GUI is also critical to facilitate the approval process for †NINGEN†. When we connect the contest system to submit the final decision, the Demo Competition Server will check the operation of system. In the competition, three †NINGEN†s : SUENAGA, SHIRAKAWA, and OGITSUKA make the final decision. We call this system KUMA SYSTEM and the team name comes from this.

4. Development Environment

Windows, MacOS X, Ubuntu / Ubuntu Server, Arch Linux.
Visual Studio 2019. Node.js, Siv3D, TensorFlow.
C++, C#, Python, JavaScript.

com-PASS (Kyogi Section)

Shuto Koikawa, Yusei Kaneyama, Yudai Okuno, Kazuhiko Nagao

National Institute of Technology, Yuge College / Japan

1. Introduction

In this competition, we play encampment game in which we can gain points by making enclosures of the tiles of the field.

2. Strategy algorithm

Our program decides where to move the agents by beam searching.

In order for the agents to avoid moving regularly, it chooses the 7 patterns which consist of the best 5 moves and 2 moves randomly chosen.

2.1 Evaluation function

We created a function based on the results of matches.

The function evaluates the condition of the field.

By using this function summarized as Table 1, the agents are able to perform proper moves.

Table 1 : Evaluation Function

	Neutral tile	Own tile	Enemy tile
Move	TP×VP	−1	
Remove (Surrounded tile)		(TP ＋ AP)×VP×1.2×(-1)	(TP ＋ AP)×VP×1.2
Remove (Common tile)		TP×VP×(-1)	TP×VP

TP = Tile points

VP = (3.5-(1 ～ 3)×0.5)

AP = Area points

2.2 Strategy we devised to win

The agents don't stay at the same place because staying at the same tile doesn't influence our tiles nor enemy tiles.

The agents don't remove tiles inside enemy enclosures because it doesn't affect the enemy's total point.

The agents don't remove our tiles which have 0 point and over.

To spread around the field, the agents don't go towards to the same position.

3. Development environment

Development OS: MacOS Mojave

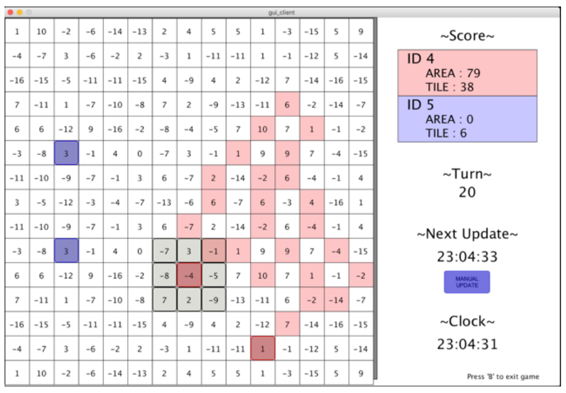
Language: Python3, Rust, C++

Library: Numpy

4. Reference

[1]Beam Search is the best DP

http://hakomof.hatenablog.com/entry/2018/12/06/000000



1	10	-2	-6	-14	-13	2	4	5	5	1	-3	-15	5	9
-4	-7	3	-6	-2	2	-3	1	-11	-11	1	-1	-12	5	-14
-16	-15	-5	-11	-11	-15	4	-9	4	2	-12	7	-14	-16	-15
7	-11	1	-7	-10	-8	7	2	-9	-13	-11	6	-2	-14	-7
6	6	-12	9	-16	-2	-8	-4	-5	7	10	7	1	-1	-2
-3	-8	3	-1	4	0	-7	3	-1	1	9	9	7	-4	-15
-11	-10	-9	-7	-1	3	6	-7	2	-14	-2	6	-4	-1	4
3	-5	-12	-3	-4	-7	-13	-6	6	-7	6	-3	4	-16	1
-11	-10	-9	-7	-1	3	6	-7	2	-14	-2	6	-4	-1	4
-3	-8	3	-1	4	0	-7	3	-1	1	9	9	7	-4	-15
6	6	-12	9	-16	-2	-8	-4	-5	7	10	7	1	-1	-2
7	-11	1	-7	-10	-8	7	2	-9	-13	-11	6	-2	-14	-7
-16	-15	-5	-11	-11	-15	4	-9	4	2	-12	7	-14	-16	-15
-4	-7	3	-6	-2	2	-3	1	-11	-11	1	-1	-12	5	-14
1	10	-2	-6	-14	-13	2	4	5	5	1	-3	-15	5	9

~Score~

ID 4
AREA : 79
TILE : 38

ID 5
AREA : 0
TILE : 6

~Turn~
20

~Next Update~
23:04:33

~Clock~
23:04:31

Press 'B' to exit game

Figure 1 ：Battle Simulator

SSBoys

Nguyen Tat Dat, Pham Quang Anh, Ho Dac Phuong

University of Science and Technology - Vietnam National University, Hanoi, Vietnam

1. Introduction

Our program is a Web application, which is developed according to MVC architecture. MVC stands for Model View Controller. But we only used 2 main components: view and controller. In the application there is a service used to calculate the best solution for the next moving steps. In that service, we use the Monte Carlo Tree Search algorithm. The descriptions of the Web application and the algorithms are as follows.

2. Web application

Our web application is written in HTML, CSS and Javascript. As mentioned above, we will describe two components: view and controller.

2.1. View component

The first is about the view component. This is the interface component for the users. Its main purpose is to show the competition field, moves and scores of the two teams. It provides the users with several functions: retrieving match information, retrieving field data, calculating the score of two teams.

2.2. Controller component

The second is about the Controller component. This is the component that receives and responds to action through the View. The main purpose of this component is to help communicate with the contest host's server. Inside there is an automated service that is executed repeatedly updates the competition field, calculates the best move and sends the move to the server.

2.3. Calculation service

In our program there is a service that is used to calculate the best possible move in each state of the competition field. This service uses a program written in C++ language to implement Monte Carlo Tree Search Algorithm.

2.3.1. Monte Carlo Tree Search Algorithm

Monte Carlo Tree Search is an algorithm using Monte Carlo Simulation method to solve many decision making and game theory problems. It uses randomness to solve problems with exponential large search space that cannot or very difficult to solve with uniform search algorithm like mini-max search.

We initialize the algorithm with an empty tree and add current field status to the root node. In each iteration we expand the tree into nodes with highest average score according to previous iterations. We also add Upper Confidence bounds applied to Trees (UCT) formula into score of each node to balance between expand the best node and searching for other promising area. After expanding the tree, we evaluate the score of the game and stored in all the ancestors of current tree node for later tree node selection. After the time limit, the algorithm will find the child node of the root node with most visit and return the move corresponding to that node.

2.3.2. Multi-agent problem

Because number of possible moves in each turn is so large for all 16 agents not the board (nearly 70 billion move combinations), we will not be able to search all those moves in given time. Instead of using a tree for all agents, we use 1 tree for each agent. In each iteration, each tree will be expanded simultaneously. The final result of each search will be back propagation in each tree using difference evaluation.

2.3.3. Difference evaluation:

After expanding the tree, we use difference evaluation for each agent's tree. The score of each agent is calculate by the difference between the current score and the score if that agent is completely removed from the system. This will help each agent to evaluate its own contributions to the system.

BKDN.AlphaGo

Hà Xuân Tiến, Ngô Tấn Trí, Huỳnh Tấn Ý, Dr. Phạm Minh Tuấn
University of Science and Technology – The University of Danang / Vietnam

1. Introduction

Our program is a Windows Form Application, which is divided into two parts: Client-side program and Server-side program. Client-side program includes a user interface which displays game field, current scores and allows the players to update the movements of agents easily, while Server-side program provides the players suggestions of next moving steps and communicate with server.

2. Client-side program

We used C# to write the Client-side program. It provides the players with several tasks such as import of competition field data, updating two teams' move, calculating 2 teams' score and competition field data backup for recovering from program failure. We designed the client with simple click events to update agents' movements. At the beginning of each turn, we receive suggestions from the server-side program and display it to the client. Players can interact with the client to edit their moves effectively.

2. Server-side program

The Server Application is written in C ++ and is run as a Windows Form Application library. Whenever the status of two teams are updated by Client-side program, the Server-side program will calculate the best solution for the next moving steps. In order to do that, we use Greedy Algorithm. Specifically, each agent will choose the best tiled so that it delivers the best results. sxn

Our current algorithm is not optimized in some cases. In the future, we will study using Minimax algorithm in combination with using multithreading to produce more optimal results in less time.

2.1 Minimax algorithm

Minimax is a kind of backtracking algorithm that is used in decision making and game theory to find the optimal move for a player, assuming that your opponent also plays optimally. It is widely used in two player turn-based games such as Tic-Tac-Toe, Backgammon, Mancala, Chess, etc.

In Minimax the two players are called maximizer and minimizer. The maximizer tries to get the highest score possible while the minimizer tries to do the opposite and get the lowest score possible.

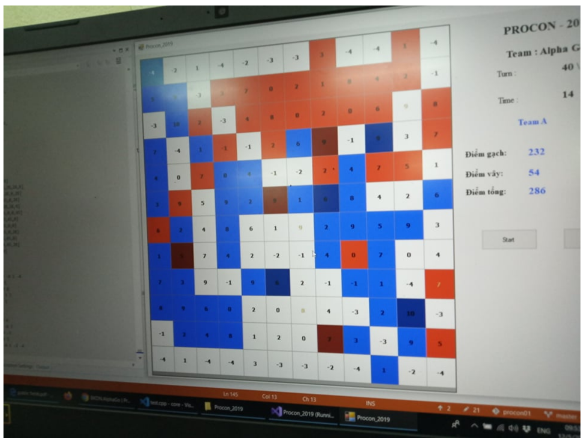
However, the complexity of the algorithm is too high. We will combine the use of multithreading to minimize runtime.

2.2 Multithreading

In computer architecture, multithreading is the ability of a central processing unit (CPU) (or a single core in a multi-core processor) to provide multiple threads of execution concurrently, supported by the operating system. This approach differs from multiprocessing. In a multithreaded application, the threads share the resources of a single or multiple cores, which include the computing units, the CPU caches, and the translation lookaside buffer (TLB).

Where multiprocessing systems include multiple complete processing units in one or more cores, multithreading aims to increase utilization of a single core by using thread-level parallelism, as well as instruction-level parallelism. As the two techniques are complementary, they are sometimes combined in systems with multiple multithreading CPUs and with CPUs with multiple multithreading cores.

3. Screen shot



Picture 1: Screen shot of the Program

Team: HaUi.TodayIFeelSoGood

Team members: Nguyen Van Luong, Nguyen Duc Linh, Nguyen Ngoc Manh

University: Ha Noi University of Industry

I. Introduction

Our program is a Web application, which is divided into two parts: Client-side program and Server-side program. Client-side program includes a user interface which allows the Tower to update the game's state easily, while Server-side program provides the Tower suggestions of next moving steps.

II. Client-side program

We used HTML, CSS, and NodeJS to write the Client-side program. It provides the Tower with several tasks such as importing of competition field data, updating two teams' move, calculating 2 teams' score and competition field data backup for recovering from program failures.
Also, Our Client-side program allows the Tower to grasp 2 teams' situation in order to decide what strategy should be used.

III. Server-side program

The Server Application was written in C++ and is run as a service on Web server. Whenever the status of two teams is updated by Client-side program, the Server-side program will calculate feasible strategies for the next moving steps. In order to do that, we use Dynamic programming algorithm, Breadth-first search and a very simple solution to choice helpful moving steps. The descriptions of the algorithms are as follows.

3.1 Dynamic programming algorithm

Dynamic programming (DP) is both a mathematical optimization method and a computer programming method. In both contexts it refers to simplifying a complicated problem by breaking it down into simpler sub-problems in a recursive manner. This algorithm helps answer a question "How many tile's points that the enemy was taken?". We option out a threshold to make up decision "Should we destroy enemy's tile's point or not?". When tile's point pass a threshold we will find the shortest possible move, in order to do that we use Breath-first search algorithm with a several changes for better solution.

3.2 Breadth-first search algorithm

Breadth-first search is an algorithm for traversing or searching tree or graph data structures. It starts at the tree root and explores all of the neighbor nodes at the present depth prior to moving on to the nodes at the next depth level. By using this algorithm, our program can find the best way to "Destroy" enemy's tile's point.

3.3 Finding helpful moving steps solutions

Our solutions just find out the highest value possible for next moving steps. The highest value here is not just choosing the biggest value. In some situations, that moving steps are to prevent enemy movement.

BKAC.Milos

Nguyen Huu Kiet, Huynh Ngoc Nhat Quang, Le Quang Tung, Tran Tuan Anh (Coach), Duong Thai Minh (Mentor)
Ho Chi Minh University of Technology, Vietnam

1. Flow program

Figure 1 is a flow chart to illustrate how to our program communicate with server. Our program sends and receives json files to proceed and draw UI to see how agents move on the map

Figure 2 is a flow chart to illustrate how to our program proceed between Backend and Frontend. We use a Turn Controller to manage the state of map, call the logic to proceed and generate move for agents and generate action.json file.

2. Strategy

Strategy brief explanation:
The strategy we tend to use is the one with simplicity and efficiency because the more complexity the tactics have the more difficulties appear afterwards. So with the intention keeping things as simple they could, the tactics applying for our strategy are easily understandable. After having all the Agents' information, we give them 2 tactics: taking points and ruining the enemies. As taking points is more important than ruining the foes, we only have one Agent grinding the opponent's gears and the others focusing on getting as many points as possible.

Those are main tactics for the Agents and that should be more than enough. Changing strategy means that we only need to adjust the number of Agents doing those tactics depend on the situation. Because of its randomness, there won't be any specific counter measure for this strategy and luck is the integral factor for every non-manual system to win against any rivals.

Let's get into the details. We'll have one Agent going zigzag in order to not let the opponent gain any area points and the other Agents going around greedily to maximize the tile points. We're not focusing on area points because it's to risky. Area points are easily obtained and turn negative points into positive ones but easily obtained means easily deleted due to the fact that Agents of other team only need to delete 1 tile to make the effort getting area points into vain. So we conclude that greedily getting all the highest tile points is the best and guaranteed way to win a match and if we're lucky we can have some area points thanks to the randomness of the way the Agents move. In some situation, moving too much is not an option. For example, there're a lot of negative tile points on the field. Moving too much would get us into worse situation as our points will reduced because of those negative points. In this case, we only need to increase more Agents that ruin the opponent's plan so that they can't have any area points and reduce the greedy Agents as moving too much is not really wise. If the field only has positive tile points, we can consider making all Agents into greedy ones so we can optimize the tile points.

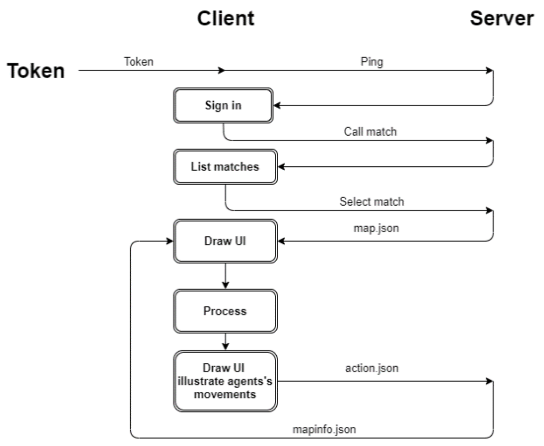


Fig 1. Flow chart of program

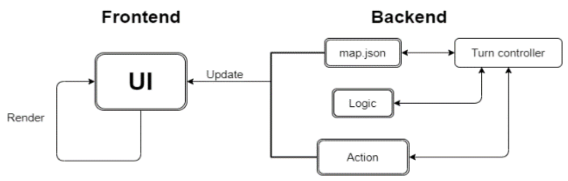


Fig 2. Backend and Frontend

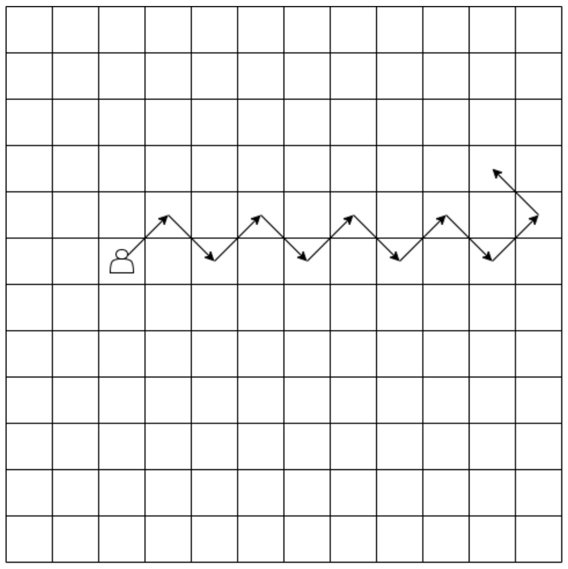


Fig 3. Path of removing tile agent

HCMUE_02 (Kyogi)

Khang Phuc Tran, Phong Van Nguyen, Trung Danh Nguyen, Quoc Thu Huu Tran
Ho Chi Minh City University of Education, Viet Nam

1. Introduction

Our program is a Web application, which is divided into two parts: Front-End vs Back-End. The Front-End part is the interface so that players can easily communicate, besides the Back-End part to help perform the remaining important operations (suggestions of next moving steps).

2. The Front-End part

We used HTML, CSS and JavaScript to write the Front-End part. It provides the Tower with several tasks such as import of competition field data, updating two teams' move, calculating 2 teams' score. It has an interface for players to observe the status of the two sides, the remaining time and the next steps can be predicted to go forward.
The basic interface consists of retrieving server information, match status: current score, number of turns, cells to move to.

3. The Back-End part

We used JavaScript to write the Back-End part. Whenever the status of two teams are updated by Front-end part, the Back-End will calculate the best solution for the next moving steps. In order to do that, we use Minimax algorithm. The descriptions of the algorithms are as follows.

3.1 Minimax algorithm

The algorithm will calculate the best cells that the player should go. Specifically, the program will calculate which cell has the highest score - that cell is the preferred location. This is a fairly simple algorithm, so during play if the player feels that the cell is not guaranteed to be the best one they can choose the more suitable one.
The program helps players choose good options. If the player does not want that option, they can choose their own option. After completing the player selection cells, just click the button to submit information.

Server TimeOut

Nguyễn Văn Bảo, Phạm Công Chiến, Phạm Thị Bích Thủy, Mai Thị Thúy Hà

Ha Noi Open University, Viet Nam

I. Introduction

Our program uses Processing and G4P (GUI for processing). The G4P library provides a large collection of 2D GUI controls. Our program has many different modes, depending on the status of the map, we can choose the most appropriate mode.

II. Agent Algorithms

Our program was written in Java and is run in Processing The program can help our to grasp 2 teams' situation in order to decide what strategy should be used. Our uses block algorithms, search algorithms and flood fill algorithms. The descriptions of the algorithms are as follows.

2.1 Block

Block is a kind of migration algorithms used in the opponent surrounding and extend the area to get regional points. Agents will move into rectangles so we use this algorithm to eat area points.

However this algorithm only moves in the rectangle, It doesn't find the way. Thus, we use search to find the best way.

2.2 Search

we need search algorithms for our program is becoming more optimal. Search can help us. Search is able to find the best solution for the next moving steps within given time. It will shorten the time for your victory. It will browse the map and save the status of all agents to find the way.

Sometime, our program may have trouble. we should prepare another plan. Hence, we wrote the flood fill algorithm.

2.3 Flood fill

Flood fill is preliminary algorithms. This algorithm is prepared for the worst case. Our program is corrupt and can not run. Flood fill will move around the enemy randomly. It is very simple and not an error. Thus, It ís a perfect B plan.

III. GUI

Display the board status and agent as shown. our team as red and our opponent as blue. Also, the score of the tile information such as whether it is positive or negative, where the enclosed area is, etc

IV. Development environment

Language: Java

Library: G4P

Revolution Technology

Nguyen Hoang Minh Cong, Le Van Thinh, Phan Luong Huan
University of Engineering and Technology - Viet Nam National University

1. Introduction

Our program is a Web application, which is divided into two parts: Client-side program and Server-side program. Client-side program includes a user interface which allows us to update the game's state easily, while Server-side program provides the suggestions of next moving steps for the agents.

2. Client-side program

We used React framework to write the Client-side program. It can perform several tasks such as import of competition field data, updating two teams' move, calculating 2 teams' score and competition field data backup for recovering from program failure. Our Client-side program can also be manually operated in case the algorithm dose not run well.

3. Server-side program

The Server Application was written in Python and run as a service on Web server. Whenever the status of two teams are updated by Client-side program, it sends a HTTP request to Server-side program in order to calculate the best solution for the next moving steps. In order to do that, we used a self-developed algorithm and Monte Carlo Tree Search algorithm. The descriptions of the algorithms are as follows :

3.1 Customize Algorithm :

Before using MCTS, we use idea of filter matrix to find high point areas and set priority for agents to take over these area.

$$P = \begin{bmatrix} & \cdots & \\ \vdots & \ddots & \vdots \\ & \cdots & \end{bmatrix}$$ P: is Point matrix.

m: is size of P

$$\mathbf{F} = \begin{bmatrix} \mathbf{1} & \cdots & \mathbf{1} \\ \vdots & \ddots & \vdots \\ \mathbf{1} & \cdots & \mathbf{1} \end{bmatrix}$$

n: is size of Filter matrix with $n = \frac{m}{4}$

v: is average value of absolute P matrix

a: is The average value of FxK matrix

K: is the matrix created by the matrix F sliding on the matrix P with the stride is 1.

$$a = \frac{1}{n2} \times \sum_{i=1,j=1}^{n} F_{ij} \times |K_{ij}|$$

```
If v > a_ij:
    Push i, j in queue. Each my agent will move to P_ij.
Else:
    Continue
```

3.2 MONTE CARLO TREE SEARCH :

In MCTS, nodes are the building blocks of the search tree. These nodes are formed based on the outcome of a number of simulations. The process of Monte Carlo Tree Search can be broken down into four distinct steps, viz., selection, expansion, simulation and backpropagation. Each of these steps is explained in details below:

Selection: In this process, the MCTS algorithm traverses the current tree from the root node using a specific strategy. The strategy uses an evaluation function to optimally select nodes with the highest estimated value. MCTS uses the Upper Confidence Bound (UCB) formula applied to trees as the strategy in the selection process to traverse the tree. It balances the exploration-exploitation trade-off. During tree traversal, a node is selected based on some parameters that return the maximum value. The parameters are characterized by the formula that is typically used for this purpose is given below.

$$S_i = x_i + C\sqrt{\frac{\ln(t)}{n_i}}$$

where:
S_i = value of a node i
x_i = empirical mean of a node i
C = a constant
t = total number of simulations

When traversing a tree during the selection process, the child node that returns the greatest value from the above equation will be one that will get selected. During traversal, once a child node is found which is also a leaf node, the MCTS jumps into the expansion step.

Expansion: In this process, a new child node is added to the tree to that node which was optimally reached during the selection process.

Simulation: In this process, a simulation is performed by choosing moves or strategies until a result or predefined state is achieved. In this phase, we did some improvements:
+ prevent agents from repeating moves, moving out of the map and go to same position.
+ calculate enemy moves
+ compare points between our team and enemy team and choose the node that has the highest probability to win (including cases of stay or removing).

Backpropagation: After determining the value of the newly added node, the remaining tree must be updated. So, the backpropagation process is performed, where it backpropagates from the new node to the root node. During the process, the number of simulation stored in each node is incremented. Also, if the new node's simulation results in a win, then the number of wins is also incremented.

SIU_Talent

Tran Van Dan Truong, Pham Thi Quynh, Van Thien Hoang
The Saigon International University / Ho Chi Minh City – Viet Nam

1. Introduction

Our team program consists of two parts: the client side and the server side. The client side is responsible for displaying the interface to help players interact more easily, thereby increasing the response speed for the game. The server side with processing algorithms is responsible for analyzing the state of the game, thereby providing the player with the most accurate steps through the client interface.

2. Client-side program

Our team used the language and technology used to design the website to create a user interface. We used HTML, CSS, Bootstrap, Javascript to write client-side programs. It provides a number of tasks, such as the position of the pieces on the board, and also the optimal move based on the server side feedback. Players will manipulate directly on the interface of this client, every move, the client will send a request to the server to process. This work is done simultaneously for multiple pieces on a chessboard at the same time, ie multitasking and multithreading.

3. Server-side program

On the server side, our team uses the C ++ programming language with object-oriented part and the Python programming language to write the application. This is an application launched under a web server service platform. When there are any state changes on the client side, those changes will be immediately updated on the server side. The server-side program will calculate the best solution for the next move to optimize the score and reduce the chances of winning by the opposing team. We have used artificial intelligence, as well as the recording and machine learning process, to handle these tasks in the simplest and most integrated way. To do that, we use the Minimax algorithm and the Alpha - Beta trimming algorithm. The description of the algorithm is as follows.

3.1. Minimax algorithm

Minimax is an algorithm that is a recursive algorithm that selects the next step in a two-player game by assigning values to the Nodes in the game tree then finding the Node with the appropriate value to take the next step. As you know, there are many search algorithms to do AI in the game such as A*, Heuristic ... Each algorithm will suit each type of game for it. The fighting games in the fighting alternately, when playing, you can develop the state space but the main difficulty is you have to calculate the reaction and moves of your opponent like? The simple solution is to assume that your opponent uses the same knowledge of state space as you. The Minimax algorithm applies this hypothesis to search for the state of the game state. The Minimax algorithm is shown by defining Nodes in the game tree: Node of MAX class assigns it the maximum value of that Node. If a node of class MIN assigns it the smallest value of that node. From these values the player will choose for himself the next most logical move. We use this algorithm to build a game tree that leads to all the available states after n steps from the current state. Our program then searches through the tree to find the best solution that can yield the most profit. However, there are too many states to be searched for in the tree. Therefore, our program takes a lot of time to process and deplete computer memory. To reduce computational and memory costs, we use Alpha - Beta trimming algorithms.

3.2 Alpha – Beta pruning algorithm

Alpha – beta pruning is a search algorithm that seeks to decrease the number of nodes that are evaluated by the minimax algorithm in its search tree. It is an adversarial search algorithm used commonly for machine playing of two-player games. It stops evaluating a move when at least one possibility has been found that proves the move to be worse than a previously examined move. Such moves need not be evaluated further. When applied to a standard minimax tree, it returns the same move as minimax would, but prunes away branches that cannot possibly influence the final decision. To illustrate this with a real-life example, suppose you're playing chess and it is your turn. You have found a good move that will improve your position. Further improvement can be achieved without sacrificing accuracy by using ordering heuristics to search earlier parts of the tree that are likely to force alpha – beta cutoffs. For example, in chess, moves that capture pieces may be examined before moves that do not, and moves that have scored highly in earlier passes through the game-tree analysis may be evaluated before others. Another common, and very cheap, heuristic is the killer heuristic, where the last move that caused a beta-cutoff at the same tree level in the tree search is always examined first.

ChromeOS

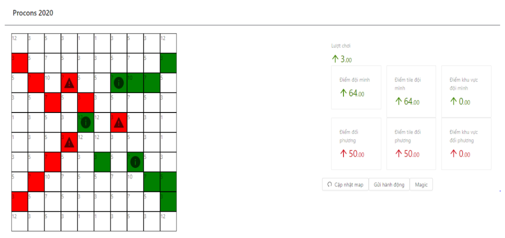
Dương Quốc Hưng, Đỗ Hoàng Khánh, Nguyễn Hải Long, Nguyễn Hoài Sơn (Coach)
University of Engineering and Technology - VNU, Vietnam

I. Introduction:

The program is a complete application, developed separately as two parts: front-end interface and back-end server. The front-end interface displays as a control panel which allows users to set up the connection with opponents, observe the match and even decide the actions of agents when users want. The back-end server is a complicate logical program, automatically calculate for the best next moves of agents in order to optimization the winning opportunity of our side.

II. Front-end Interface:

The front-end interface was programmed with ReactJS, React-dnd, Redux, and RxJS. This part of the application handle several tasks including importing data of the match, visualizing them for users to watch with the status of the match is updated continuously. A noticeable feature is that despite the next movements are majorly chosen by the back-end server, users are still able to make decisions manually by a few simple interactions.



Screenshot of front-end interface

III. Back-end Server:

We virtualized the web server of the contestants in order to process Kyogi games without depending on the judged system.

We implemented our competitive strategy as a web service in Java. This back-end part will communicate with the front-end side through Rest API to receive the on-going result of the match, do the calculation to find the most effective movement in the case and then return the result to the front-end as well. At the moment, we have been using 3 different algorithms for this part, training by letting them duel against the other and adjust the value of parameters in both automatic and manual methods. The ultimate algorithm we shall use in the competition will be chosen flexibly.

The two algorithms are:

A. Local Valuation Method:

- Agents are considered one by one.
- At first, we have a positive integer K (K is chosen manually to be smaller than 5). The original table will be called the K-th layer. Each square on the layer will be evaluated by the point of its neighbor squares through a complex function. That value will be set for each available K-step movement (the possible set of movements which can be acquired in next K turns by a single agent), set maximize on the destiny square to create the (K-1)-th layer.
- The process is redone iteratively as K decreases. When the value of K reaches 1, the program can determine the best choice for the current agent through the final table. The set of movements promoted by that choice will be tracked and blocked on corresponding layers.

B. Global Valuation Method:

- All agents are considered at the same time. We calculate the best value for each agent to reach each square on the table, based on the intrinsic point of squares on the paths multiply to a weight of distance (the further distance, the smaller weight).
- After that, these values and their track are put into a system of linear equations. Solving the equations using a few optimization algorithms shall give an effective set of movements.

C. Min-cost flow Method:

- All agents are considered at the same time. In each move, we match each agent with each different tile using the Min-cost flow algorithm and building an evaluated system for agents' move . Then, we choose the best matching state which help to earn the most points.
- The advantage of this algorithm is that not only we do not need to concern about the collision states of agents because of each agent will be matched with different tiles, but we can also find the area tiles of the opponent which is detected by the evaluated system.

By combining flexibly 3 above strategies, we can access that which strategy is preferred for a given situation.

Plan Y

Le Duc Tung, Nguyen Phu Truong, Ho Dac Phuong (Coach)
University of Engineering and Technology, Vietnam National University, Hanoi, Vietnam

1. Flow program

At first, we get TOKEN from the server and pass it to our program as an input. A few seconds before the match start, we keep sending request (get match) to server until it actually starts. Then we handle json file to get data of the match. When the algorithm yields output, we post the action.json file and wait for the next turn. Our program is separate with the visualization part. We write a small tool to get data from API, pass to our program and post data to the server.

For visual purposes, we write a html file using Bootstrap to draw the map, and a simple javascript to ping, get map.

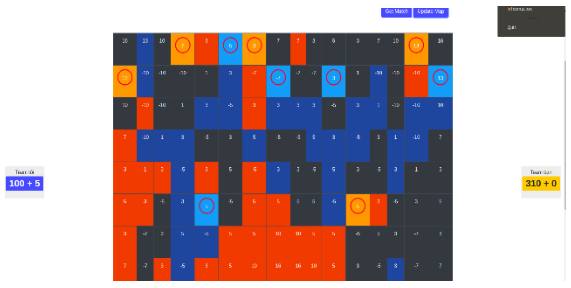


Fig 1. The visual

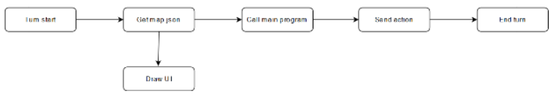


Fig 2. Program flow

2. Strategy

Strategy brief explanation:
Because the more complex the tactics are, the more difficult it is to debug the code and upgrade the strategy, we tend to use a strategy that is simple while ensure it is as efficient as possible. So with that intention in mind, the final tactics applying for our strategy can be easily understood. At the beginning of each turn, after getting all the information of the game, we give the Agents 2 options: taking tiles for points or ruining enemy's area points. We have an algorithm to evaluate the benefit from ruining the enemies for each Agent, if it does not give enough benefit, that Agent will keep taking tiles instead, as taking points is more important.

Those are main tactics for the Agents and that should be more than enough. The Agents will automatically choose their act depend on the situation, so we only have to change the benefit level when we need to adjust the strategy. Because the enemy never knows the current benefit level and our way to judge, it is hard for them to find a specific strategy to counter this, and luck is the integral factor for every non-manual system to win any games.

Let's get into the details. We're not focusing on area points because it's too risky. Area points can be easily obtained and have a great impact on total points since it can turn negative points to positive ones. But "Easy come, easy go", they can be easily removed by opponents, due to the fact that enemy Agents can remove just a tile and make all the advantages and effort vanish into the void. So we mainly focus on greedily maximizing the tile points and hope we are lucky to get some area points from the randomness in the way Agents moving around. We prefer to send the Agent moving zigzag in order to not let the opponent gain many area points and hopefully get some area points as there may be some tiles between the line. At some places where the zigzag lines give negative points, we will have the Agents move in straight lines. As for ruining enemy area points, we detect all enemy bounds, find the best way for our Agents to remove them, and judge the benefit before making any decision.

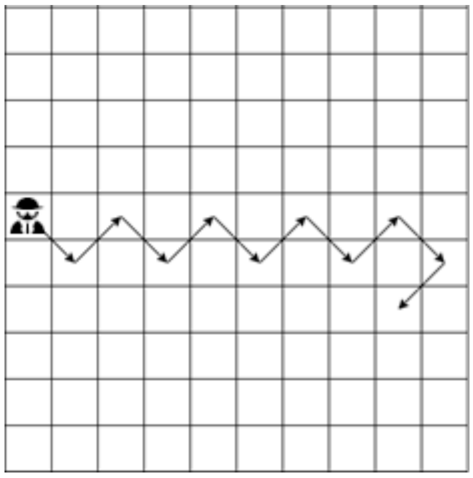


Fig 3. The zigzag line

Fibo

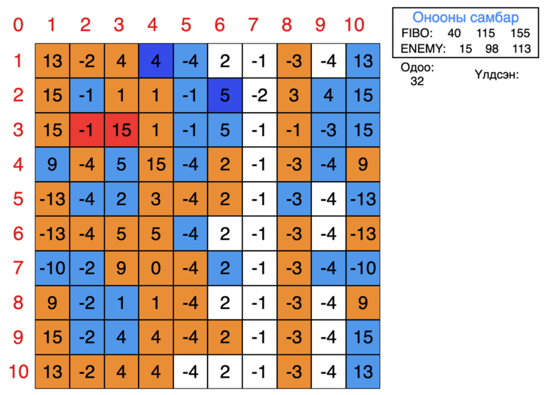
Lkhagvadorj Darjaa (Student), Tuguldur Bayarsaikhan (Student), Dalaijargal Purevsuren (Mentor), Dorjnamjirmaa Badraa (Mentor)
National University of Mongolia, Mongolia

1. Introduction

Our system will use Node.js, which is a web-based technology, to retrieve input from the server, and returning the data. It is the most widespread runtime with other good qualities, which is why we have chosen it.

2. System description

Specifically, our main algorithm BackTrack processes the data, from the field, with the help of HTML and Javascript. The time we are given depends on the number of moves left and the number of agents on the field, so every move is handled with a different strategy that is considered to be the best during its respective turn.
The whole game is divided into 3 parts according to the number of moves, and in each part, the agents have tried to vary the distance and the strategy of the game. An illustration of our system is presented in Fig. 1.



0	1	2	3	4	5	6	7	8	9	10
1	13	-2	4	4	-4	2	-1	-3	-4	13
2	15	-1	1	1	-1	5	-2	3	4	15
3	15	-1	15	1	-1	5	-1	-1	-3	15
4	9	-4	5	15	-4	2	-1	-3	-4	9
5	-13	-4	2	3	-4	2	-1	-3	-4	-13
6	-13	-4	5	5	-4	2	-1	-3	-4	-13
7	-10	-2	9	0	-4	2	-1	-3	-4	-10
8	9	-2	1	1	-4	2	-1	-3	-4	9
9	15	-2	4	4	-4	2	-1	-3	-4	15
10	13	-2	4	4	-4	2	-1	-3	-4	13

Онооны самбар

FIBO:	40	115	155
ENEMY:	15	98	113

Одоо:
32

Үлдсэн:

Fig. 1 Illustration of our system

VTC Engineering (Kyogi Section)

Cheuk-Yin Chan, Ho-Lam Man, Cheung-Sing Yiu, Chi-Pang Lam
Vocational Training Council - Pro-Act Training and Development Centre (Electronics), Hong Kong

1. Introduction

This year, in the Competition Section, we will conduct a type of a territory game to fight over which team can take more squares on a field divided up into grids.

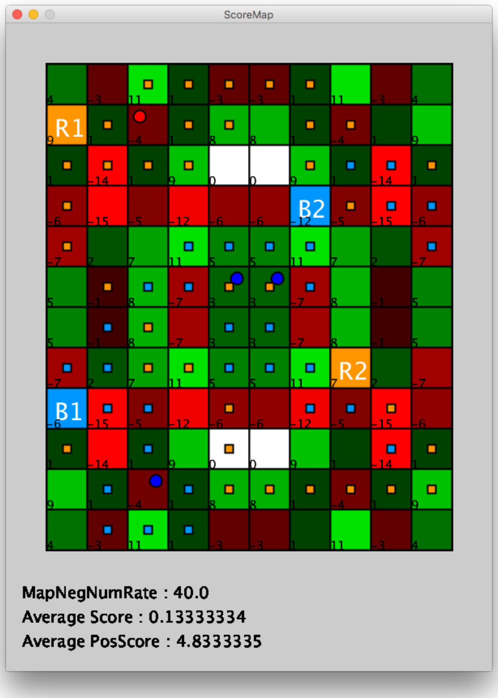
2. Overview of the system

2.1 Setup of the system

Our system requires 3 laptop computers. All laptop computers will connected to the site network through a hubs/switch. The first computer will use as a communication note, to receive the field data and send out the command to server in order to move the agent. The rest of 2 laptop computers are used for strategy planning.

2.2 Introduction of System

We had developed a set of tools to help us to determine which strategy we are going to use. Like the "Score Map Tool", it can be used to display the score in different color, If the score of the square is tend to +16, the color of background will tend to light green, and vice versa.



MapNegNumRate : 40.0
Average Score : 0.13333334
Average PosScore : 4.8333335

Fig.1 The Score map of the application

2.3 Overview of the algorithm

We had added artificial intelligence (AI) to help us for making optimal decisions.

Monte Carlo Tree Search (MCTS) combines the generality of random simulation with the precision of tree search.

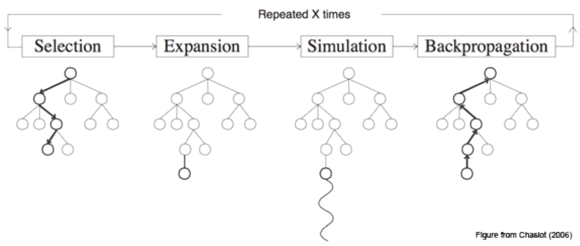


Fig.2 Monte Carlo Tree Search (MCTS)

Selection - Starting at root node *R*, recursively select optimal child nodes (explained below) until a leaf node *L* is reached.

Expansion - If *L* is a not a terminal node (i.e. it does not end the game) then create one or more child nodes and select one *C*.

Simulation - Run a simulated playout until a result is achieved.

Backpropagation - Update the current move sequence with the simulation result.

2.6 Apply MCTS

We had apply the MCTS concept into our program, we search all the possible steps node by node, and found out the optimal decisions, but there are one limitation we are facing, the search tree is too huge, so we are going to trim down the size of the search tree, so that we can boost up our searching performance.

Themed Section

:::doc (tendoc)

Ryuta Itabashi, Haruki Fujimaki, Saito Kadowaki, Soichiro Suzuki, Yoichi Kamoshita, Akihiro Yamashita
Tokyo National College of Technology, Japan

1. Introduction

When a blind person wants to read something printed on the paper, they had to rely on "Braille Translating Service", which often contained in city hall or so. But it costs a lot, and it takes quite long.

So, we developed a system called ":::doc" that can translate any document to braille and print automatically and is operated by some very simple interface.

2. Overview

":::doc" has three distinctive functions.

First, the system can translate any documents to braille instantly. This function will help blind people to get information from documents which contain some information they want to know immediately such as advertisement of discount in a nearby supermarket, an announcement of classroom visitation in user's kid's school, and user's apartment's electricity bill.

Second, the system can also translate in another way, braille document to typically printed text. This function will help people who work at school for blind people to read something that students wrote.

Lastly, the system can send the braille data to another ":::doc" system by scanning the braille document. Because this function is quite similar to the FAX, we call it a "Braille FAX". This function helps blind people who are far away to communicate with each other.

These all functions can be controlled by user's voice using Google Home Mini, or by a simple button array.

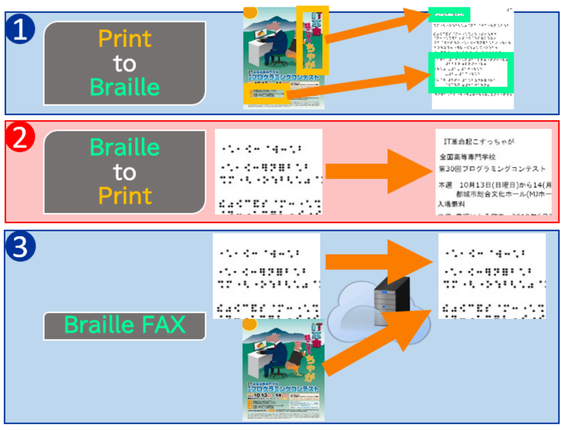


Fig.1 Functions overview

3 Structure of the system

We use an Intel Compute Stick as a central controller of the whole system. Then we attached an EPSON MFP, to scan documents and print the standard documents, a braille printer made by company SINKA to print braille documents, and a Raspberry Pi 3B to communicate with these printers.

Also, we used a 3d-printed original button array and Google Home Mini as an input interface.

The system is simple and clean, easy to use.

4 Additional features

When translating a document which has long sentences to braille, the system has to use more paper than original printed document because Braille documents have quite less information density. This will waste the paper, and takes longer to read the translated braille.

So we added a text summarization feature which uses RNN Network. This function will summarize the long sentences and focus on some crucial information.

This feature is still experimental, but works quite good, outputs very accurate sentences.

5 Conclusion

With our system ":::doc", the blind people no longer have to rely on any paid Braille translating services.

They can get any information immediately with some seriously simple and easy operation. So now, they can save their money and also their precious time.

This system will make communication between non-blind and a blind person more active and will make everything in the daily life of blind people smoother and faster.

Toba Map; Plotting Citizen Information on a Map

-Visualization of Future and Past Data, Consideration and Evaluation for Public Facilities-

Student: Tsuyoshi Takahashi, Marin Yamaguchi, Yuki Kamaya, Riku Tsuji, Yuito Nakanishi
Mentor：Nobuo Ezaki
National Institute of Technology, Toba College, Japan

1. Introduction

In recent years, there has been the less birthrate and more aging of the population in Toba City, Mie Prefecture. In 1986, There were 28,000 population in Toba, but in 2019 the population has decreased to 18,000. Toba city needs to think about public facilities they hold. The city hall has various information, though it is not being used effectively. Therefore, we cooperated with Toba City and made a system to connect Basic Resident Register with public facility location information to plot on a map. We will propose "Toba map," a web application for examining the current problems and the future situation.

2. System Overview

"Toba map" can filter the various data plotted on the map by attributes and it is a system that can analyze optional data.

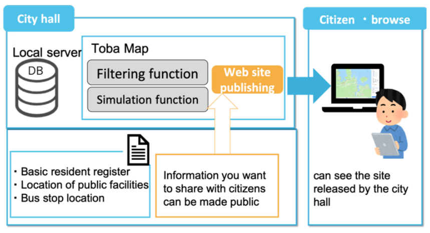

City hall
Local server
DB
Toba Map
Filtering function
Simulation function
Web site publishing
Citizen ・browse
• Basic resident register
• Location of public facilities
• Bus stop location
Information you want to share with citizens can be made public
can see the site released by the city hall

Figure 1 System Configuration Diagram

3.Feature Description

3.1 Residents / House plots

The basic function of this system is to plot the date of the inhabitants and residences data. It is possible to put some remark on the location of residents on the map according to their ages and genders. You can learn more about the distribution about some areas with many elementary school students, others with many senior citizens. By using the time slider, you can view the future, past, and present distribution of the residents in Toba City.

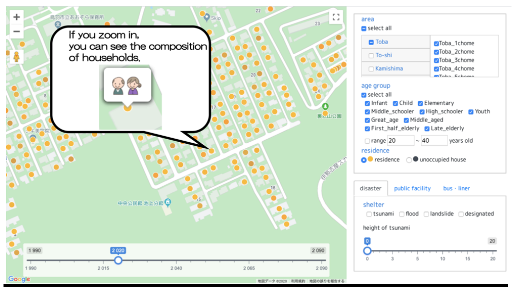


Figure 2 Plot Screen of Residents and Dwellings

3.2 Analysis of Usage of Public Facilities

This system can visualize the number of people and age groups at the public halls and bus stops, and calculate the evaluation values from various perspectives. In addition, facilities can be moved by dragging or can be pseudo-erased. The evaluation value is recalculated according to this movement. Using this value, it is possible to consider the integration or movement of the facilities. For example, this system can simulate about inundation prediction by Tsunami. When Tsunami happens, this system can visualize what kind of age group it is and how many citizens come to the Tsunami shelter. And when Tsunami actually happens, you can realize how many supplies are necessary.

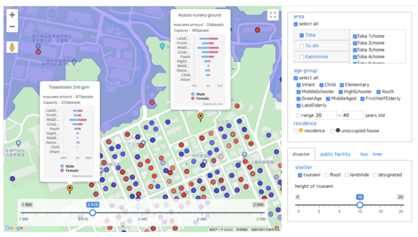


Figure 3 Example of Plot and Score Display of Public Hall

3.3 Broadcasting Live image about liner

Toba has five remort islands, where citizens live there. They usually use liners to move from their islands to the main land. We have constructed a system that can get and stream about the liner location informaition and its live image. While checking the live image, the liner staff can make a cancellation decision. Marine information may be shared between the seafarers and officers.

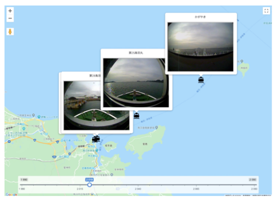


Figure 4 Liner Location Information and Live Image

4. In Conclusion

We have cooperated with Toba City and developed this system while sharing information closely. The analysis results will be publicly available not only to the city staff but also to the citizens. Everybody will be able to make an access to this system and the public information to make Toba city much better.

Search-a-BLE(Kadai Section)

Serina Ito, Yuka Koyama, Kazuki Kishida, Kazuhiko Nagao
National Institute of Technology(KOSEN),Yuge College /Japan

1. Introduction

In Japan 2018, about 4 million items were losted.
It becomes 3.8 billon Yen($ 38 millon US Dollar).
In the old days of Japan, communication was active and people cooperated each other.At now, there is no room for people helping others.Several dictation system for lost items were developed, which uses BLE(Blue Tooth Low-Energy) and Smart Phone.These systems are developed for the specific devices, They collect the user information, the location and the items.The information is related to privacy, and it should not be accessed easily.We propose the lost item dictation system “Search-A-BLE” that does not require the server and specific devices.

2. Several search method

“Search-A-BLE” is the lost item dictation system, which uses any BLE devices. A User attaches a BLE device on the item he doesn’t want to lose.The user can monitor BLE devices by smartphone.We do not use any server for collect data, and we consider privacy problems.Figure 1 shows a system configuration diagrams.

2.1 Alart for lost dictation

When the connection between the BLE device and smartphone is lost, the smartphone to announces this to the user by alart message.Since the location and dates were also recorded on the smartphone, the user can search the last founded areas.

2.2 Colavolative search

“Search-A-BLE” collects the peripheral BLE device ID, location and date, and stores them in the smartphone for some periods.When the user passed the other user, search request (which includes lost ID and date) are exchanged via BLE connection.This data exchange mechanism is called D2D(device to device) transfer protocol.
This search request is relay many users by owner’s activities.If a smartphone finds the target, it creates the discovery information and relay to other smartphones. Finally, the owner who lost item is notified like (as follow;) “your item was found at xxxx and time”.The finished search request is also relayed to other users.It depends on the users’s activities and communications.

3. The features of this system

3.1 Support any BLE devices

In BLE 4.0, the devices should have not-specified IDs (the MAC address and UUID) for any access and devices.
For this reason, many similar systems use the specified BLE device only.In the preliminary experiment, we discovered that the IDs can be specified by combining multiple values.These values can be obtained when established a connection between the smartphone and devices.With this identification method, “search-A-BLE” becomes the lost item dictation system for any BLE devices.

3.2 Consideration of personal information

“Search-A-BLE” collects several information, BLE device IDs, the locations and the owners.　It is necessary to treat the personal information appropriately.In the system, all acquired data are encrypted and stored only in user’s own smartphone.Since the user is not linked to the location, there is no infringement of personal information.Since no server is used , information leak risk can be completely prevented.

4 Conclutions

We developed the lost item dictation system using any BLE devices and D2D transfer protocol.
By not using a server, it is possible to solve the information leak problem, and to search even in disasters or no radio waves.By using this system, people would feel other’s consideration.　We hope that enhanced the system helps to develop the community.

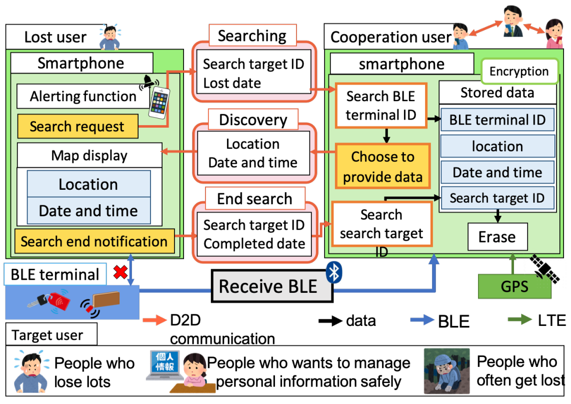


Figure 1: system configuration diagra

ANIMAL CAPTURE -AR Game For Wheelchair Users-

Satsuki Okuda, Yuki Matsumoto, Toshiki Mizuta, Ryo Fujiwara, Tsuyoshi Hashimoto
National Institute of Technology, Matsue College, Japan

1. Introduction

Wheelchair users have limited types of play compared to non-handicapped people.　According to a report, more than 50% of wheelchair users do not play any recreations.[1] Furthermore, it causes difficulty of communication between wheelchair users and non-handicapped people. We visited a local special education school and asked students about their school life and activity. Many students said that they like to play video game. But according to teachers, some students cannot play video games because they cannot move their body freely by themselves. The story was very impressive for us. Therefore, we developed a game for wheelchair users using gaze input, "ANIMAL CAPTURE".

2. Overview of ANIMAL CAPTURE

"ANIMAL CAPTURE" can be played by a wheelchair user who wears HoloLens. The player can get point to feed hungry animals. The screen of HoloLens is shared on outside tablet. Game watchers can see the state of the game. Furthermore, player and game watchers can communicate by telling the position of hungry animals.

2.1 Game Flow

I. Looking for hungry animals.
II. Approach hungry animals to move wheelchair.
III. Gaze hungry animals to feed them. The player gets points.

If bait runs out, the player have to go to the bait box and pass over the box to increase bait. (Fig.1)

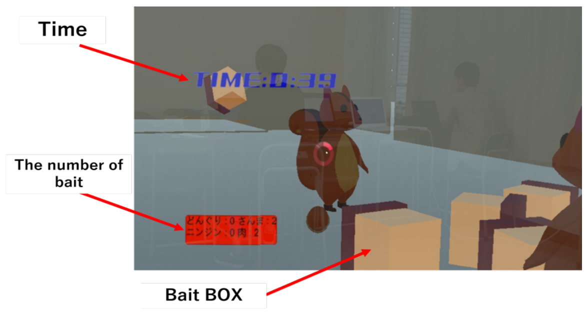


Fig.1 The game screen

2.2 Equipment

The game field is room such as a classroom. Players use wheelchair and HoloLens. Game watchers see display to share the screen of HoloLens. The game also needs wireless LAN router to share the screen.

2.3 System Constitution

The game models were made using Blender. The game were made by Unity. Player's position is gotten by SLAM function of HoloLens.

2.4 Detail of Gaze Input

When the distance between player and animals is short, player can use gaze input toward animals. When the player matches the white pointer of HoloLens with hungry animals, a gauge appears in front of the animal. When the gauge become full, the player can feed the animal. (Fig.2)

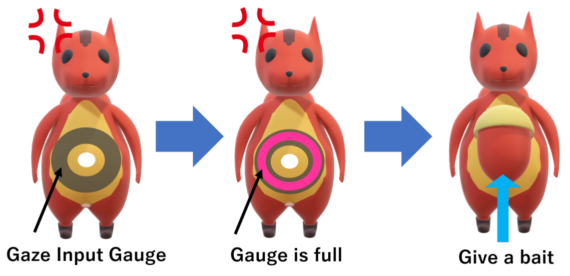


Fig.2 Flow of gaze input

3. Perspective of Regional Activation

Anyone can play the game. This is an advantage of this game. In fact, when we visited the special education school, students enjoyed to play the game with us together. We and wheelchair users could communicate through the game. We also exhibited the game at the open source conference in Shimane prefecture. Men and women of all ages could play the game there. The game makes communication among many people. "ANIMAL CAPTURE" can contribute to the regional exchange.

[1] "Basic data of sports for handicapped people", Ministry of Education, Culture, Sports, Science and Technology, June 2015

uniHome

Hokama Rui, Irie Hiroki, Bise Kisato, Kishimoto Rin
National Institute of Technology, Okinawa Collrge / Japan

1. Introduction

In Japan, the number of families living in different areas for each of the three generations has increased. Less contact through phone calls or e-mail with family living away weaken family ties. Therefore, we offer a system called “uniHome” that sets a robot in each home and can give a sense of family presence even how far they are through.

2. System overview

“uniHome” uses a human-like robot to make it approachable. It offers “automatic connection between long-distant family members” and “Daily sense of family presence.” These give you a sense of security.

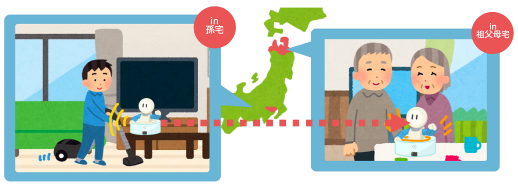


Figure 1. User image

3. The communication robot “unibo”

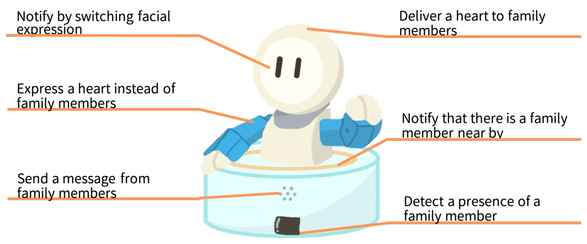

Notify by switching facial expression
Deliver a heart to family members
Express a heart instead of family members
Notify that there is a family member near by
Send a message from family members
Detect a presence of a family member

Figure 2. The functions of “unibo”

“unibo” tells and delivers your message and heart by simple motions. For example, if you come close to “unibo,” it can automatically send your presence and the heart which thinking of your family living away. Also, the LED lamp on the robot’s board lights and tells grandparents that their grandchild is by the robot in real-time.

4. 3 Functions of uniHome

1. Function to Show your presence by being by the robot

A unibo in each family members’ home shows you and your family that somebody is by the robot. It offers a daily family presence even if family members are living away.

2. Function to express your heart by rubbing the robot

The robot shows your family that you are thinking of them by dancing and deliver the warmth of home.

3.Function to greet

The robot greets by posing certain poses for each greeting so that you can feel connected with the family living away.

5. System configuration

Figure 3 shows a system configuration diagram.

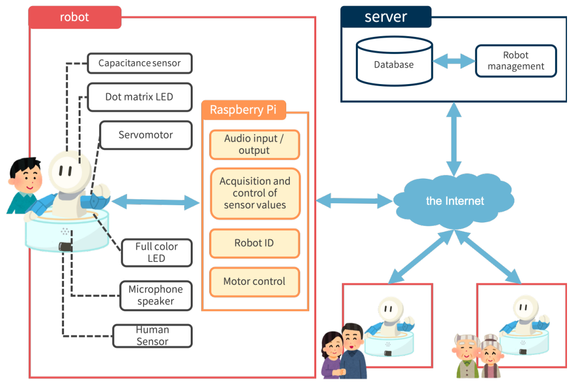
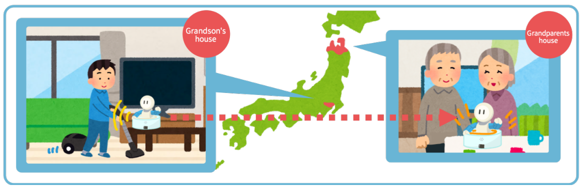


Figure 3. System configuration diagram

6. At the end

By “uniHome,” we will tighten family ties and connects between long-distance family members.



Agricowture: Near-future cattle grazing support system

Shogo Hirose, Sorata Sagi, Kotaro Higuchi, and Shuichi Fukunaga
Tokyo Metropolitan College of Industrial Technology, Japan

1.Introduction

In recent years, a system that can take care of livestock is required because the livestock industry is facing a labor shortage due to aging. We suggest a system called "Agricowture" that can monitor and manage the condition of cows even at a distance by using drones, apps and GPS devices.

2. System Overview

With Agricowture, the user can check the location of a cow using apps by attaching the GPS device to the cow, and then the user can check the state of the cow by a drone. Figure 1 shows a schematic diagram of the system.

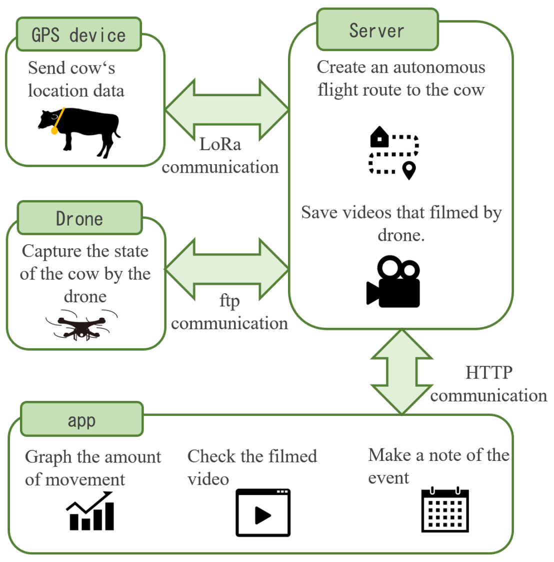


Figure 1. Agricowture system overview

2.1 GPS devices

The GPS device for getting the cow's location information can be set up by just hanging it on the cow's neck. Transmitting and receiving GPS data uses LoRa WAN, which has low power consumption and can be used by anyone.

2.2 Check cow's health

In Agricowture, the location of a cow with a GPS device is displayed on the map of the app by a marker (Figure 2, left). The user can set a name for each marker. The app also allows you to check the amount of cow movement on a graph. (Figure 2, right). Cows fidget when they are in estrus, and the amount of movement increases because they move greatly from the location of the herd. We use that to detect estrus. When cow estrus is suspected, the color of the marker displayed on the map changes from red to yellow.

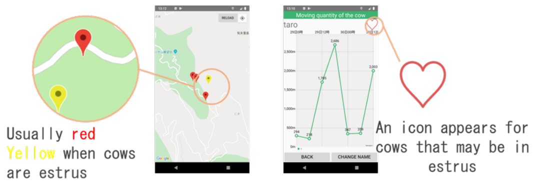

Usually red
Yellow when cows are estrus
taro
Moving quantity of the cow
BACK
CHANGE NAME
An icon appears for cows that may be in estrus

Figure 2. Displaying cow status (left) and graphical display of travel distance (right)

2.3 Check the image of the cow

When there is a cow that is suspected of estrus, you can fly a drone to the cow and check a filmed video with the app. The drone can be flown by tapping the marker of the cow which you want to check the status of. (Figure 3). Anyone can easily take videos with a drone because the drone heads to the cow by autonomous flight. The video taken by the drone can be viewed from the app, so you can see the cow without going to where it is. By watching the video, the user can schedule events such as childbirth, and put them on the app's calendar.

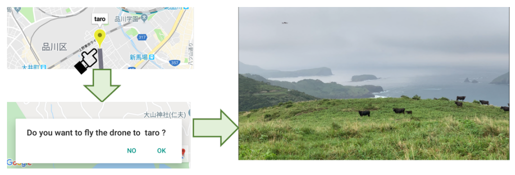

Do you want to fly the drone to taro ?

Figure 3. Flow of shooting with a drone

3. Conclusion

We are repeating field tests with the cooperation of ranch farms in Shimane prefecture, Japan. The realization of this system will contribute to the development of the livestock industry.

Naprock Procon – Access Denied

Trần Mạnh Hùng, Bùi Văn Hùng, Đặng Thị Hương Lan, Nguyễn Đình Tưởng
Hanoi Open University, Vietnam

1. Introduction

Recently, there have been many fires in our whole country, causing serious consequences on the property and human life. These are not only agencies, factories, and flammable plants, but also in our home where we live in, there are always potential risks. The reason was determined from various sources, including some things that may be simple but with very serious consequences.

The sensors systems can measure sudden changes in temperature in the air. Not only for this reason, but sensors systems can also do things that people can't do. It can be used to measure factors such as air quality, the leaking gas during cooking, and sensor radiation can also detect harmful substances. It is useful that these sensors can warn of a fire hazard and notify the owner or manager of the business to find out many appropriate preventions.

2. Sensor system's description

The hardware devices use for Sensor system's development:

• Using a microcontroller as a control center.

• Input device: receive information where the fire occurred and transmit its signals to the fire alarm center.

- Smoke sensor for direct monitoring detects smoke signals.
- Air quality sensor monitors directly to detect signs of dangerous toxic gas.
- The temperature sensor checks the temperature of the environment within the protected area when the temperature of the environment does not meet the initial set requirements, it will send an alarm signal to the processing center.
- The fire sensor detects ultraviolet rays emitted from the flame.

• Output device: Receive a signal from the fire alarm control panel.

- Fire alarm: Installed at the processing center, with the function of sounding an alarm signal in case of an incident, to notify those around them for a solution, timely evacuation.
- Light: symbolizes the processing stations, has an alarm function and helps supervisors can identify the location of the incident.
- Sim module is used to send a signal to notify the manager to ensure the urgency.
- Pre-installed preventive measures including automatic nozzles and ventilation fans.

3. Conclusion

This is a topic that is receiving great attention from society. With the desire of the research team to build a system that can be applied well and widely in practice. The sensor system will help people to detect early and promptly prevent fires. Along with that, minimize installation costs and equipment design to be more widely applicable to everyone.

Optimization of Lighting Control Systems in Smart Homes for Energy Savings

Xuan Viet Cuong Nguyen, Minh Hoang Ngo, Quang Khai Duong, Xuan Anh Do,
Hoai Son Nguyen (mentor)
VNU-University of Engineering and Technology, Hanoi, Vietnam

1. Introduction

With the high-speed development of computer and network technologies, a new paradigm of Internet of Things (IoT) that things around us such as RFID, sensors, electrical devices, etc. can connect to the Internet, which gradually becomes a reality. Smart homes are among widespread IoT applications because they can bring to users a comfortable living environment. Smart lighting system is one of the most basic and essential systems for smart homes. Our system enhances the user's experience and life quality, especially for young children and the elderly, who might have trouble in moving or using smart devices to control indoor lighting system. The positions and behaviors of the users will be captured by cameras, then the system will analyzed the data and send the control signal to the lighting system. In additional, the main advantage of our system is to minimize the total energy consumption of the whole system. This is a very important issue and is concerned by all countries around the world. By integrating an appropriate algorithm to control light, our smart lighting system contributes a great deal of energy savings as well as economical to users.

2. System Description

Our proposed system contains 3 main components: The hardware part inside the home, the server part and the user part (Fig. 1).

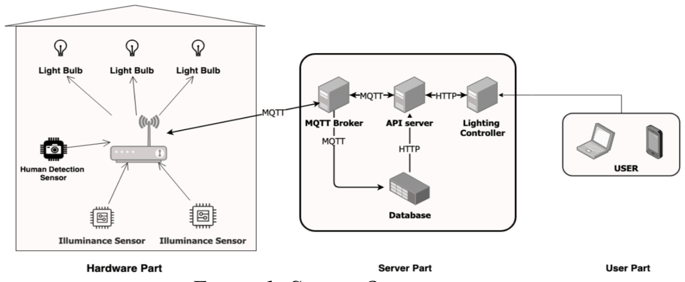


Figure 1: System Overview

The hardware part in the home includes light devices, human detection sensors and illuminance sensors. These devices communicate with a special device called Home Gateway by a network protocol such as Zigbee or Echonet Lite.

These server elements can be located on cloud servers and shared with other smart home services.

The last component is the user part, which is a mobile app to monitor and control the smart lighting system inside their house. Sometimes, the desired brightness level calculated by the lighting controller based on use activity does not meet a user's desire. In that case, users can use Mobile App to adjust the desired illuminance level at their desire locations. The Mobile App also directly connects to the server part to receive information about smart lighting system.

3. Results

We have built a prototype smart lighting system and installed the system within a plastic model house. We performed a number of experiences on our testbed environment to verify the operation of our system. We evaluate the difference between user desired brightness level and sensing illuminance controlled by our smart lighting system. The experimental results of 5 testcases are shown in Table 1. The value -1 of a lighting area means that there is no request at that lighting area. The camera system after being integrated will act as a device to automatically provide the user's location in the room. After detecting if there is a person in the room, the camera will try to identify user's location, behavior and map it to the corresponding location on the light map in the room, then send information into server in order to get appropriate light turning pattern.

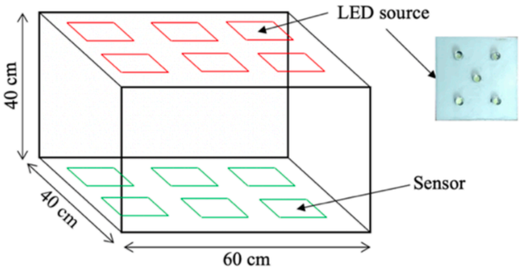


Figure 2. Prototype house design model

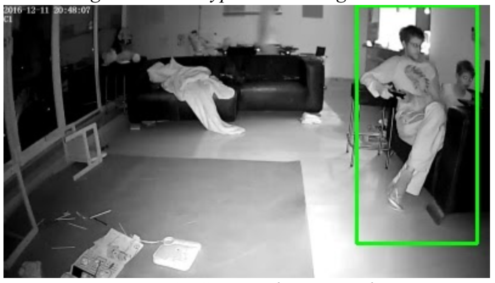


Figure 3: Camera detect people

Testcase	No. of requested position	Requested illuminance value at each position (lux)	Actual brightness at each position (lux)
1	1	100, -1, -1, -1, -1, -1	112, 24, 23, 24, 4, 4
2	2	120, 150, -1, -1, -1, -1	137, 144, 49, 49, 9, 9
3	3	50, 80, 100, -1, -1, -1	50, 72, 106, 34, 15, 15
4	4	50, 80, 100, -1, 85, -1	57, 72, 92, 68, 81, 27
5	5	20, 120, -1, 30, 150, 10	30, 125, 50, 55, 165, 35

Table 1: Results when system runs in real cases

4. Conclusion

The system has solved several the disadvantages of current smart lighting systems in the market. The system is operated based on an intelligent light control algorithm - (GA - Genetic Algorithm). This control algorithm is calculated based on the values from the illuminance sensor which will give the best light turning pattern to meet the desired user brightness while still minimizes electric energy consumption. With this light control algorithm, the indoor lighting system can both maximize the energy from nature illuminance and use the energy consumed for the system appropriately. Moreover, we have integrated the camera to detect user behavior so that the system can control the lights according to the user's behavior, helping the system operate as automatically as possible.The lighting system consists of devices developed according to Echonet Lite, and we have built a complete API and web and mobile app to help users and developers more easily add Other systems later to have a complete IoT Platform.

VIRTUAL REALITY (VR) IN FOLKLORE

Nguyen Thi Linh, Bui Khanh Ngoc Anh, Lai Thi Thu Phuong, Ph.D Ma Thi Chau
VNU University of Engineering and Technology, Viet Nam

1. Introduction

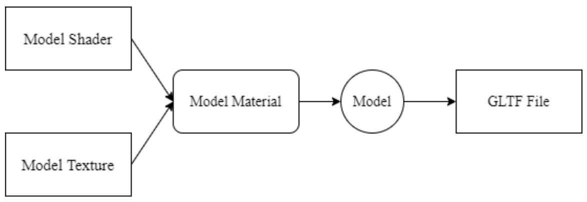
In the modern world, the development of technology has contributed to simpler and more convenient life. However, there are still not many applications to support people to have closer contact with folklore. This makes the folklore is gradually disappering in the awereness of young people nowadays. Water puppet Vietnamese folk art is disappering gradually. This art use puppets controlled by artists to act on the water surface. To make water puppet can come closer to people, "VR in folklore" is created. The product can help people know the culture of water puppet honestly and easily with the virtual reality technology used on the web.

2. System descriptions

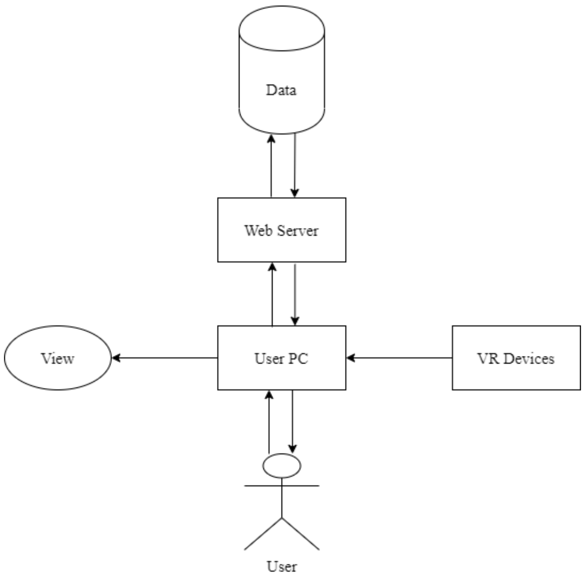
Programming on the web using A-Frame (a web framework for building virtual reality experiences) makes it access to product easily. User only needs a computer connected to internet to have a perfect experience. Furthermore, using A-Frame also meets the different VR platforms and devices like: HTC Vive, GearVR, Rift,… Basically, "VR in folklore" is easy to use, maintain and develop.

Create and operate product:

- Build virtual reality experiences on the web by A-Frame.
- Making data: The data includes 3D models and audio. From real enviroment, create 3D models by Blender software, then export the format to a GLTF file (format files for 3D models and animations).
- User uses a computer connected to VR devices (or internet) and experiences it in a virtual enviroment. This environment is related to water puppetry art.



The model asset workflow.



General architecture of webVR application

3. Conclusions

In conclusion, our product provides people a simple and easy way to experience the culture of water puppet Vietnamese by using virtual reality technology. Therefore, it can be developed as an intangible museum with many different art forms by virtual reality, contributing to preserving and developing traditional Vienamese culture.

InnoVAte : The Smart irrigation system

Cao Quyết Thắng, Nguyễn Văn Hiếu, Nguyễn Hoài Sơn (Mentor)
University of Engineering and Technology, Vietnam National University, Hanoi

1. Introduction

At present, the agriculture of Vietnam are backward and the growth slows down compared to the other sector. The water irrigation, especially, faces a lot of problems and difficulties in the climate change period. Meanwhile, information technology is growing strongly and its applications are appearing in many fields. Agriculture is no exception. The application of information technology for farming will help farmers to have a more convenient and efficient way to farm and increase product quality as well as contribute to the current agriculture modernization. We have designed a smart irrigation system called InnoVAte, which manages the farms and gives a watering strategy more efficiently.

2. System descriptions

Our irrigation system is designed to manage and automatically control watering devices in the plots based on information collected from sensors and a weather forecast service. Our system has three components:

- Sensor nodes: collect information of air environment and soil moisture and send to Gateway by Lora communication.
- Gateway: manages sensors nodes and watering devices and transfers sensor data between these devices and Server by MQTT protocol.
- Server: a computer which stores and processes farm data to give a watering control strategy based on Model predictive control (MPC). Our watering control strategy optimizes the amount of water required for irrigation and get the most out of the natural water source such as rainwater.

We have implemented our system as follows:

A. Devices:

- Sensor nodes: include soil temperature and soil moisture sensor DHT10, air temperature and humidity sensor AM2301, light sensor BH1750FVI. Besides, it has a LoRa SX1278P 433MHz module to communicate to Gateway.

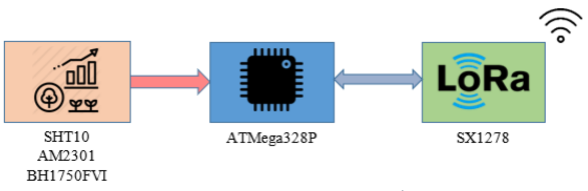


Figure 1. Sensor node

- Gateway: a Raspberry Pi 4 connected to a LoRa SX1278 433MHz module via ATMega328P microcontroller, which is used to communicate with sensors and pump devices. The Raspberry Pi communicate with Server through MQTT protocol.

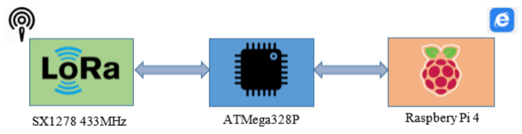


Figure 2. Gateway

- Watering devices: a pump and drip irrigation nozzles. We also use LoRa SX1278P 433MHz module for this node to communicate with Gateway.

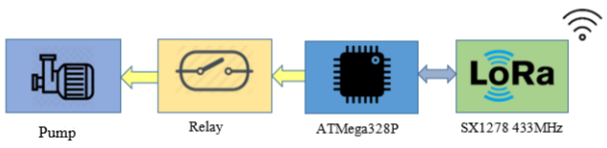


Figure 1. Controller node

B. Server: uses Model predictive control (MPC) to make a watering decision. The soil moisture in a future period is predicted based on a soil moisture model and the weather forecast information. The amount of water required for irrigation at each watering period is optimized based on the prediction of the soil moisture in the period.

C. Network Platforms: The system uses LoRa network for connecting sensor nodes, control nodes and Gateway on the farm. We design a protocol running on LoRa platform to make communication between devices easier and manage them more efficiently. Gateway and Server are communicated via MQTT protocol on the Internet.

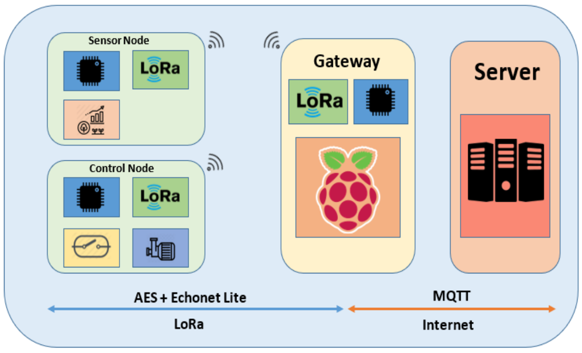


Figure 2. Smart irrigation system architecture

3. Conclusions

To summarize, our system provides an automated solution for water irrigation activities on the farms. It helps the irrigation task for farmer to be easier and more efficient. Future, we will improve the system to make the system easier to use and increase the efficiency of the whole system.

A Novel Framework G-DANs for HADR (Kadai Section)

Jabez Tho, Hans Delano, Teo Shin Jen
Singapore Polytechnic, Singapore

1. Introduction

Typhoons are formed when the ocean surface heats up, evaporates and form clouds that can translate to strong winds and heavy rain under the right environmental conditions. Typhoon becomes destructive when it enters populated land. We proposed a novel framework the Generative Destructive Adversarial Networks (G-DANs), a rapid analysis and prediction method on satellite imagery built upon CycleGAN to simulate the aftermath of typhoon at landfall and predict destruction hotspots. This information will allow non-governmental organizations the equipment required for debris removal to search for survivors after the disaster.

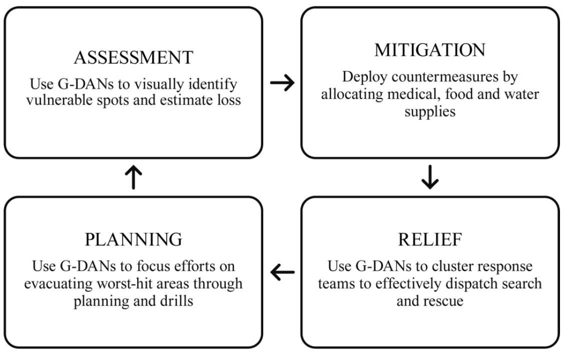

ASSESSMENT
Use G-DANs to visually identify vulnerable spots and estimate loss
MITIGATION
Deploy countermeasures by allocating medical, food and water supplies
RELIEF
Use G-DANs to cluster response teams to effectively dispatch search and rescue
PLANNING
Use G-DANs to focus efforts on evacuating worst-hit areas through planning and drills

Fig. 1 Humanitarian assistance and disaster relief management (HADR) using G-DANs

2. Methodology

Our novel framework G-DANs built upon a deep learning neural network model CycleGan, which is an extension of Generative Adversarial Networks (GAN). GAN works by pitting 2 neural network systems against each other to improve the quality of their results.

To train the model for G-DANs, we process before and after satellite images from the worst-hit city of Tacloban in the Philippines during Typhoon Haiyan (Fig. 2a and 2b). The model extracted unique characteristics from unpaired pre-typhoon and post-typhoon images for image translation. To improve the robustness of our model, we input other satellite images of destruction sites with a 1:2 ratio where we had 100+ post-typhoon images and nearly 300 pre-typhoon images. We trained the model until the loss curve had converged and the images generated became more representative of a typhoon destruction path after almost a hundred epochs.

3. Observations

G-DANs training satellite images taken from the Philippines, Tacloban in 2013(Fig. 2a and 2b). Evaluation with the use of satellite images from Singapore, Choa Chu Kang as the input to G-DANs (Fig. 2c) The generated image (Fig. 2d) depicts the simulated destruction hotspots and aftermath.

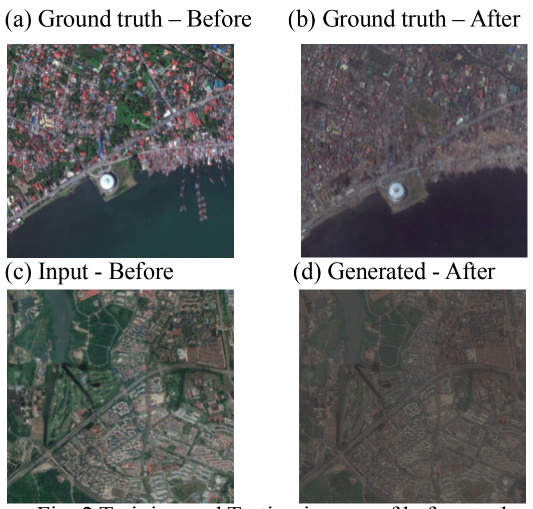


Fig. 2 Training and Testing images of before typhoon landfall and after typhoon landfall

4. System Diagram

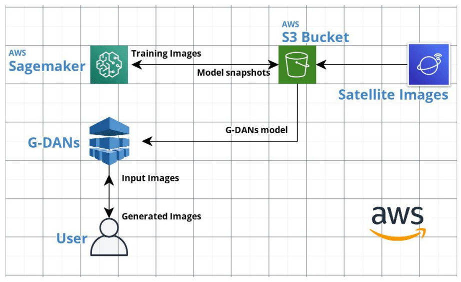


Fig. 3 Utilising cloud computing to train G-DANs

5. Roadmap and Improvements

Our novel framework G-DANs received constructive feedback from the personnel from Changi Regional HADR Coordination Centre (RHCC), Defence Science & Technology Agency (DSTA), Airbus and Singapore Space and Technology Association (SSTA). We intend to tune the model with more satellite images from different regions and to add in additional parameters such as typhoon direction and wind speed to improve the accuracy of the model's predictions. G-DANs is open-sourced, and accessible through: https://github.com/Jasperabez/G-DANs.

TelMed Care

Nafisah Kamaruzzaman, Noor Fatin Najwa Othman, Nur Amirah Yahaya &
Assoc. Prof. Dr Sakinah Ali Pitchay
Universiti Sains Islam Malaysia (USIM)
Malaysia

ABSTRACT

Controlled medicine is crucial during traveling and TelMed Care is proposed to integrate technology in daily life and to increase the accessibility rate of the medicine for civilians. Internet of Things (IoT) based on teleportable medicine for traveller (TelMed) is a healthcare product that offers online consultation with doctors via mobile apps which integrated with vending machine. It provides medical online services and instant solutions to enable user take better care of themselves. It uses cloud connection to communicate with vending machine that located in limited access such as boarding area in airport and train station. Registered user will be notified via sms after transaction have been made through apps and QR code will be generated for obtaining the controlled medicine at preferable vending machine. It offers security element where the QR code is valid in one-time purchase only. According to travel behaviour study and increasing pattern of traveller, two main problem statements have been identified which are: (i) patient tends to forget their medicine when travelling and (ii) less accessibility to doctors and restricted medicine in limited access area. Our IoT concept for e-Health services support the smart cities application which integrates communications of 5G and employs cashless payment. Thus, IoT based teleportable medicine features help to solve the society needs by increasing the health of traveller instantly by supplying controlled medicine with doctor's prescription without any delay which support of emerging technologies.

Keywords: Traveller, Portable medicine, Healthcare, IoT

1. INTRODUCTION

According to travel behaviour study by G.Musa and O.F Sim (2010) and due to increasing pattern of traveller, two main problem statements have been identified which are: (i) patient tends to forget their medicine when travelling and (ii) less accessibility to doctors and restricted medicine in limited access area specifically in boarding area such as at the airport and main train station. There are two main components in this proposed solution which are TelMed Care apps and TelMed Care vending machine. It has two users such as registered patient and doctor/admin.

2. OBJECTIVE

TelMed Care aims to provide a convenient platform of virtual prescription and an accessible medicine vending machine that located at limited boarding area. This application will integrate technology in daily life where Wi-Fi connection is required. Besides, TelMed Care aims to increase the accessibility rate of medicines for civilians. The app features provide instant prescribed medicine and location of TelMed Care vending machine.

3. NOVELTY

The novelty and inventiveness of proposed solution:

a) TelMed Care is developed to facilitate the traveller that registered under this application.
b) TelMed Care is an IoT based teleportable medicine for traveller in healthcare field that offers online consultation with doctors via mobile application which integrated with vending machine.
c) TravMed Care supports emerging technologies and fits the smart cities concept.

4. SECURITY FEATURES

a) Provides multi-factor authentication where user is required to scan QR code on vending machine and enter their identification no. before the medicine is released. User will also be notified via sms.
b) In terms of medicine safety, online prescription of controlled medicine by traveller only for the registered patient to the specific hospital via TelMed Care apps.

5. USEFULNESS

This section is expected to aid specifically to the traveller.

a) Instant and offers effective method on controlled machine for general prescribed medicine.
b) Reduce time and cost to obtain the medicine without going to the hospital or pharmacy while traveling.
c) Avoid cancellation of appointment with doctor when the date laps with travel time.
d) Towards cashless teleportable medicine.

In terms of commercial value, it has a high commercial value among the traveller who needs controlled medicine instantly. The automated process will speed up the process of obtaining the required medicine. It can be marketable to any international airport boarding or train station area since Wi-Fi connection can be accessed easily.

6. CONCLUSION

TelMed Care aims to facilitate traveller that requires controlled medicine in urgent situation by providing instant online communication via mobile apps. The medicine can be obtained based on the chosen location.

7. REFERENCES

G.Musa and O.F Sim (2010). Travel behaviour: A study of older Malaysians. Current Issues in Tourism, Vol. 13, No. 2. March 2010, pp. 177-192.

DocIX: Document Information Extractor

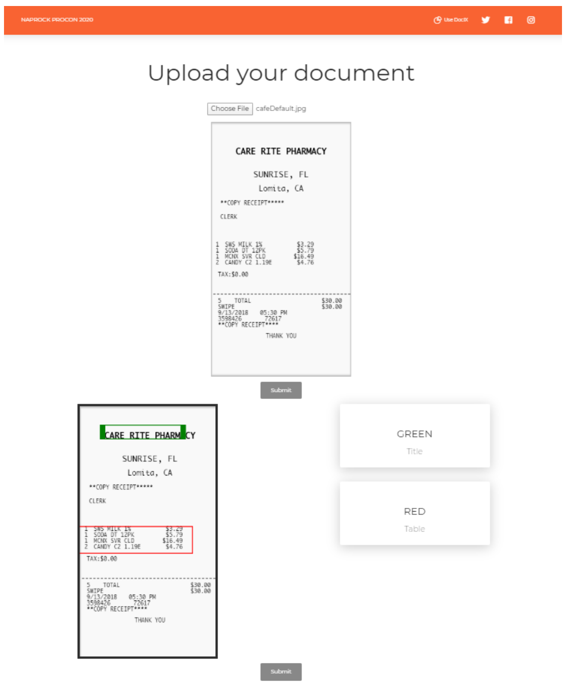
Nunja Sowan, Waruwat Chaiyadith, Kitsuchart Pasupa
King Mongkut's Institute of Technology Ladkrabang, Thailand

1. Introduction

Nowadays, documents are everywhere, for example, a receipt. Transactions happen in almost every second around the world, and most of them generate some kinds of document. A receipt contains many pieces of useful information such as transaction list, issue date, address and a lot more. These pieces of information can be useful for enterprise resource planning and supply chain management. For example, an accountant in a company must collect receipts from employees who want to receive their reimbursement from the company. Gathering these pieces of information manually is laborious and errors might be easily occurred. Currently, computer vision has many powerful features such as Optical Character Recognition (OCR) which can be used to extract text from scanned or photographed documents. Taking advantage of these features, we propose a system called "DocIX: Document Information Extractor" to assist humans to collect the data within a document. The proposed system utilizes computer vision and machine learning. The performance of this system may not be better than that of a human in the beginning. Therefore, in order to achieve a higher performance, we let users improve our system by correcting the inaccurate predictions. This approach simultaneously improves the performance of the system while also aiding humans.

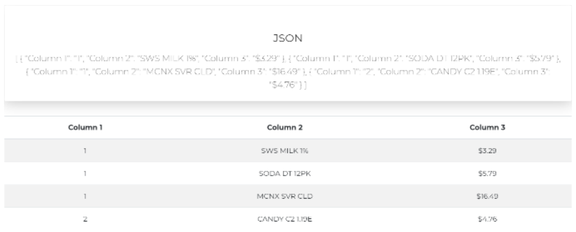
2. Features

The proposed system can detect the information area in a document. Users can upload their scanned or photographed document into our system. After it has been uploaded, bounding boxes of detected areas of information appear automatically, as shown in Fig. 1. Since there may be some errors during the information area detection process, we allow users to reposition and resize bounding boxes as they want. They also have to confirm the position and size of the bounding boxes. The correct coordinates are stored in our database which will be used for improving the performance of the system. Any text within each bounding box area will be extracted by OCR into a JavaScript Object Notation (JSON), as shown in Figure 2. Users can freely set the keys and values of the JSON as they prefer.



```
CARE RITE PHARMACY

SUNRISE, FL
Lomita, CA
**COPY RECEIPT*****

CLERK

1  SWS MILK 1%        $3.29
1  SODA DT 12PK       $5.79
1  MCNX SVR CLD      $16.49
2  CANDY C2 1.19E     $4.76

TAX:$0.00
------------------------------------
5    TOTAL                   $30.00
SWIPE                        $30.00
9/13/2018   05:30 PM
3598426      72617
**COPY RECEIPT****

THANK YOU
```

Fig. 1 User Interface



[{ "Column 1": "1", "Column 2": "SWS MILK 1%", "Column 3": "$3.29" }, { "Column 1": "1", "Column 2": "SODA DT 12PK", "Column 3": "$5.79" }, { "Column 1": "1", "Column 2": "MCNX SVR CLD", "Column 3": "$16.49" }, { "Column 1": "2", "Column 2": "CANDY C2 1.19E", "Column 3": "$4.76" }]

Column 1	Column 2	Column 3
1	SWS MILK 1%	$3.29
1	SODA DT 12PK	$5.79
1	MCNX SVR CLD	$16.49
2	CANDY C2 1.19E	$4.76

Fig. 2 JSON Output

3. Algorithm

Our proposed system consists of the following techniques. (i) Image Processing: Several techniques have been used to improve the overall performance of our system, for example, Hough Transform for line detection; (ii) Information Area Detection: Regions with Convolutional Neural Networks (R-CNN) is used for information area detection. This model can detect several kinds of objects such as human, and vehicle. In our case, the area of information in a document is detected; (iii) OCR: Google's Tesseract OCR is used to recognize an image of a text and produce the text itself.

4. System Requirements and Tools

(i) Python 3.7.x; (ii) Node Package Manager v6.x; (iii) A server with an NVIDIA graphic card.