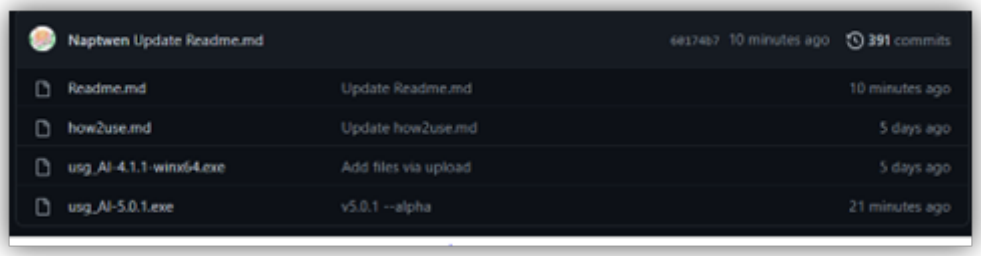
USG AI ENGINE USER GUIDE

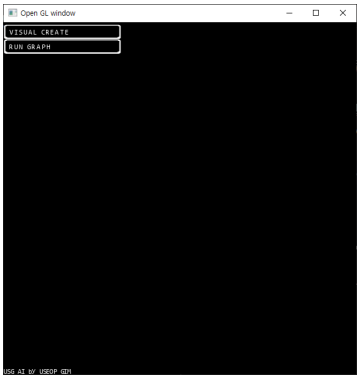
The source code of this program is created by Useop Gim and the license is following the GNU AFFERO LICENSE V3 with third party licenses for each OpenGL, OpenCL, OpenCV, base64, FreeType Please check for more detail in the attached license file.

How to intall

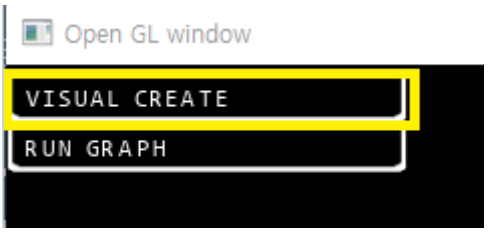
1. For working on the GUI and GPGPU. the pre-installing is required for the OpenGL 3.0 (for GUI interface), OpenCV 2.0 (for testing), OpenCL 2.0 (for gpgpu calcualtion)
2. Access the https://github.com/Naptwen/usgAI then download the newest version



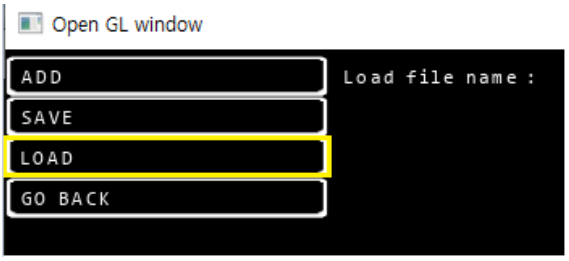
3. After installing for the test, double click the usg_AI.exe file

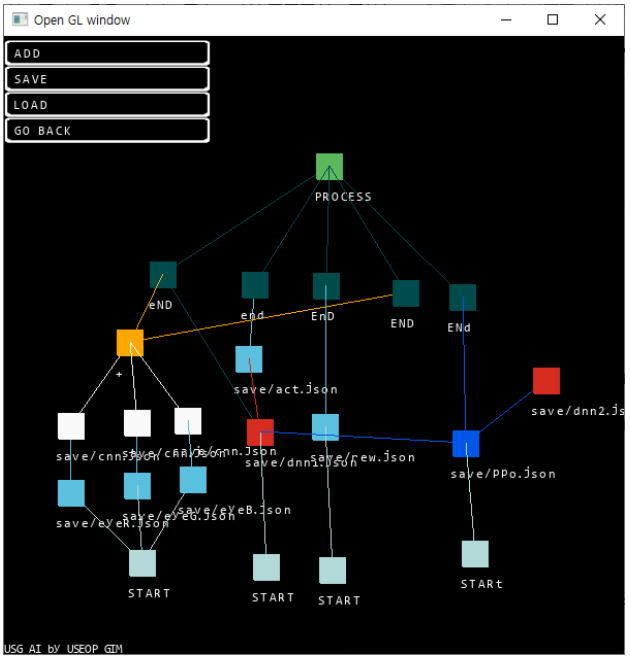


4. For testing, click the VISUAL CREATE button



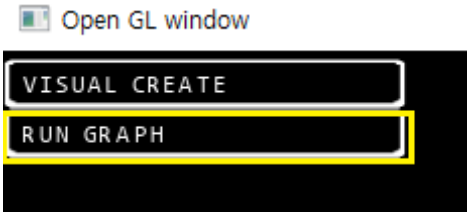
5. Next click the LOAD button then type “save/test.text’”



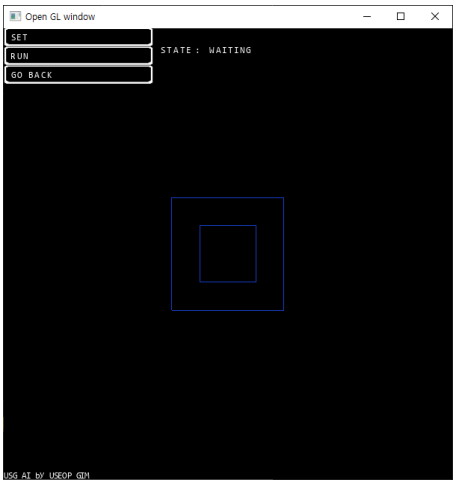


- If you can't see like above, please re-installing the file or checking the save folder in usg_AI

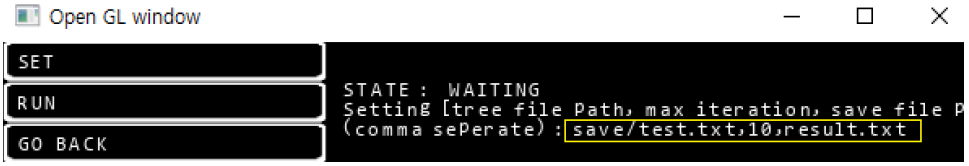
7. Click the GOBACK button then click the RUN GRAPH



8. In the RUN GRAPH screen, click the SET button



9. After click the SET button type “save/test.txt, 10, result.txt”

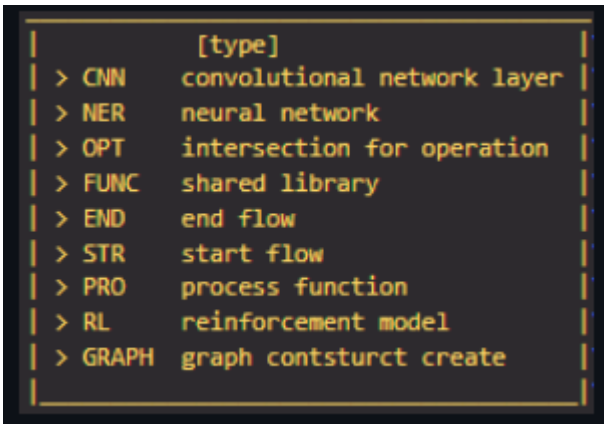


10. Now click the RUN button
11. If the center double square is moving, it means that the programe is running.
12. If the moving is stopped, you can exit the program

Console Interface User Guide

Although the console interface is provided, you can write down by other third party text edit program. In this case, please reference the format that already in the save file.

--help : show the basic options
--license : show license
--create “type”: writing the model using console interface



CNN : Convolutional Neural Network block
NER : Deep Neural Network block
OPT : Operation block (version 5 provides only summation and multiplication)
FUNC : The shared library block (shared library block is a extra function for runtime)
END: The end of one cycle of flow graph block
STR: The start of one cycle of flow graph block
PRO: The control the cycle(process) of the flow graph block
RL: The reinforcement model block
GRAPH: The graph flow chart file

CNN OPTIONS

```
|        < CNN LAYER LIST >        |
| > PADDING          [PA]          |
| > SOBEL_X          [SX]          |
| > SOBEL_Y          [SY]          |
| > SOBEL            [SB]          |
| > IDENTITY         [ID]          |
| > EDGE             [ED]          |
| > SHARP            [SH]          |
| > BLUR             [BR]          |
| > RELU             [RL]          |
| > MAX POOLING      [MP]          |
| > AVG POOLING      [AP]          |
| > SUM POOLING      [SP]          |
|__________________________________|
```

- *All paddings are size 1*
  PADDING : padding the image basic setting is zero padding.
- *All kernels are 3 by 3 with stride 1 (v5.x.x version don't allow to change it)*
  SOBEL_X : Sobel mask X axis basic setting
  SOBEL_Y : Sobel mask Y axis basic setting
  IDENTITY : Identity mask
  EDGE : Edge mask
  SHARP : Sharp mask
  BLUR : Blur mask
- *All poolings are 2 by 2 with stride 2*
  MAX POOLING : max pooling
  AVG POOLING: average pooling
  SUM POOLING : summation pooling

** When the allocated image by stride size is not enough the allocated part is discarded.*

Neural Network options

```
< FUNCTION LIST >
> LINEAR FUNCTION      [X]
> SIGMOID FUNCTION     [S]
> ARCTAN FUNCTION      [A]
> LEAK RELU            [L]
> RELU                 [R]
> ZNORMALIZATION       [Z]
> MIN MAX NORMAL       [M]
> SOFTMAX              [S]

< INITAL FUNCTION LIST >
> XAVIER INIT          [X]
> LECUN INIT           [L]
> HE INIT              [H]

< OPTIMA FUNCTION LIST >
> NONE                 [DI]
> ADAM                 [AD]
> NADAM                [NA]

< COST FUNCTION LIST >
> KL-divergence        [KL]
> Mean Square          [MS]
> Surrogate            [SR]
```

Active function and normalization function

LINEAR FUNCTION : f(x) = x

SIGOMID FUNCTION : sigomid (x)

ARCTAN FUNCTION : arctan(x)

LEAK RELU : f(x) = max(x,0.3)

RELU : f(x) = max(x,0)

ZNORMALIZATION : z-norm function

MIN MAX NORMAL : min max norm function if min == max it return 0

SOFTMAX : softmax(x)

Initial function

XAVIER INIT : xavier uniformal initialization

LECUN INIT : lecun uniformal initialization

HE INIT : He uniformal intialization

Optimization

NONE : no optimization for back propagation

ADAM : adam optimization for back propagation

NADAM : Nestrov adam optimization for back propagation

Cost function

KL-divergence : KL divergence cost function

Mean square : mean square with power of 2 function

Surrogate : surrogate cost function (it means depends on the model's cost evaluation)

```
|          < RL MODEL LIST >   |
| > DQN                [DQN]   |
| > PPO                [PPO]   |
| > SAC                [SAC]   |
```

PPO: Clipping policy based reinforcement learning model.

DQN: The first google reinforcement learning model.

SAC: The stochastic actor critic reinforcement learning model.

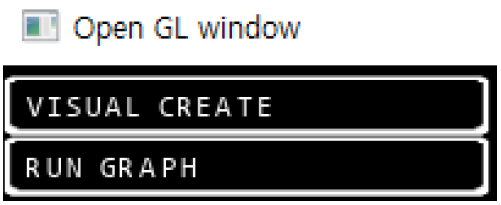
--setting : Changing the setting options

verboseon(off) : Show the detail of the program runtime message

gpgpuon(off) : Using OpenCL GPGPU

--run "graph file path" "max iteration" "save file name" : running the program

GUI User Guide



VISUAL CREATE : Creating the graphical flow chart
RUN GRAPH : Running the graphical flow chart

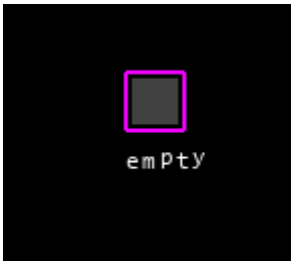
VISUAL CREATE GUI Guide

First of all, the block is the unit for saving and running each algorithm
This program's one of main point is that using the connections of blocks for easy to create the machine learning algorithm.

ADD : Adding a new block
SAVE : Save current selected block and its child blocks
LOAD : Loading the graph flow chart

Guide for using graphical interface

1. When click the ADD button, empty block is created.
   The total node is limited in 2^32-1
2. When click the block, the selected block is highlighted.



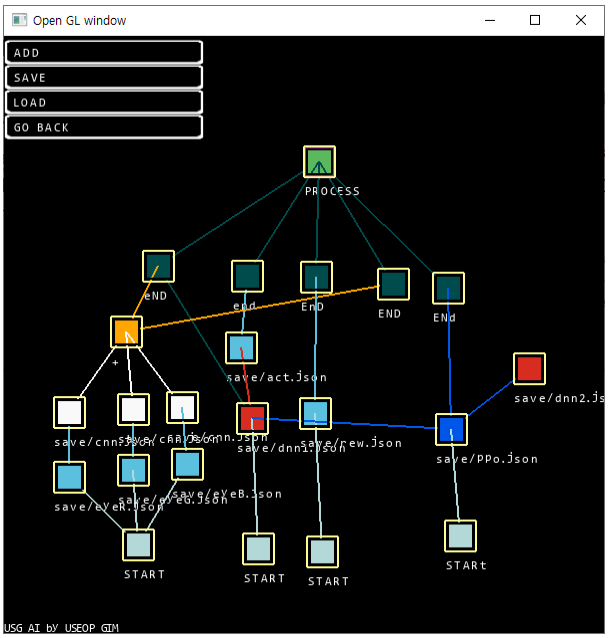
3. When right click the block, you can link the block with file by file name

4. For each file is colored by its type(model)

Especially, file name as “+”,”x”, “END”,” START” are automatically created even there are no such file.

5. From selected block to another, the first block be child and the second be parent for the selected block

6. When click the connected block, it also highlighted the selected block and all others.

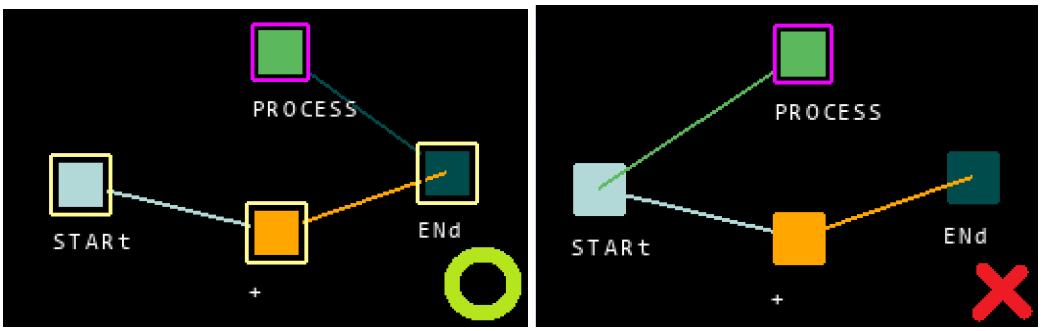


7. When double click the selected block, its connections are removed.

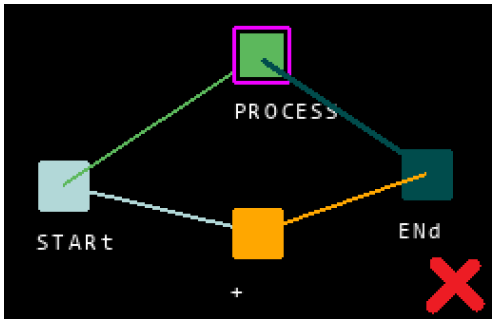
User Guide for connection

1. The START and END block must be exist at the end of both side in the block that want to be run.

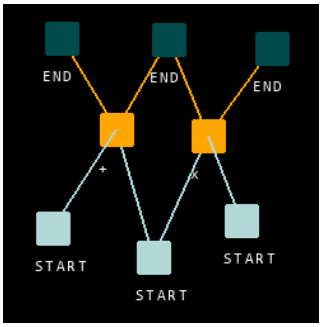
2. The PROCESS block must have the END block as child.



If the connection be recursive, the program is forced to be exited!

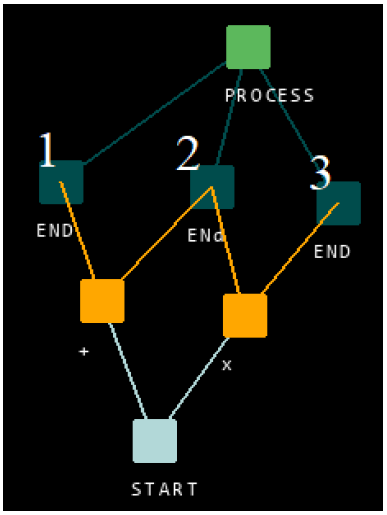


3. There are unlimited for the number of child and parents

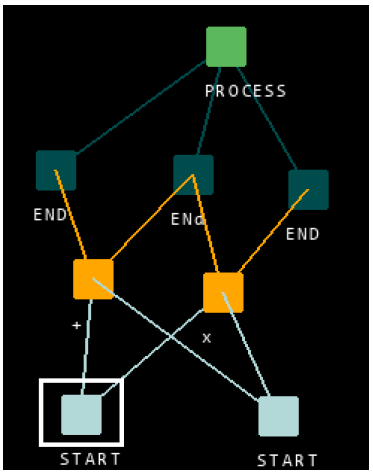


When save the block, all connected node are saved together but not disconnected block

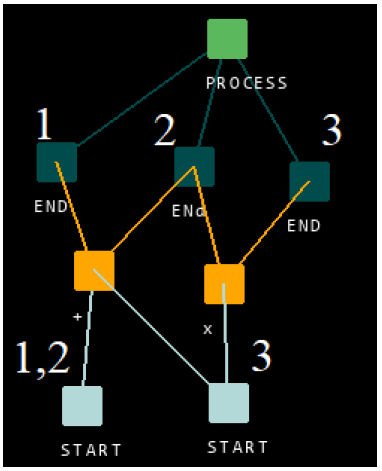
4. The order of running the program is base on the position of end block connecting PROCESS block. The most left upside block be front.



5. As same as above, the order of the START block is also the most left upside block be first.



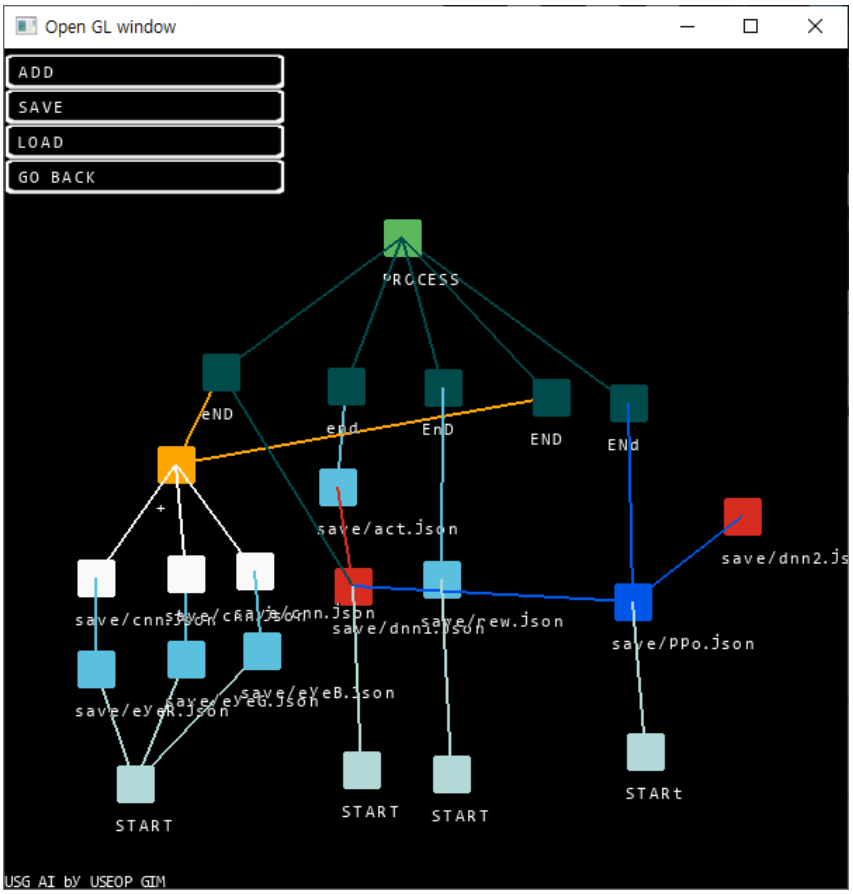
For example, above connection shows only the left START block be the start line for all END blocks.



Beside, above connection shows the 1,2 END block is start from the left START block and 3 END block is start from the right START block.

How to make Reinforcement block

The reinforcement block must have a single PROCESS block with four END block and multiple neural network block (the number of neural network block depeneds on the reinforcement learning model)



- END [1] : the first end block recieve the enviroment state.
- END [2] : the second end block recieve the action state.
- END [3] : the third end block recieve the reward state.
- END [4] : the last end block recieve the enviroment state after the action.
- The most left side neural network block be the main agent block for the learning, action(or policy) in model. In other words, it is the main neural network decides the action.

More detail of structure.

1. Get the display area pixel RGB data seperately by eyeR,eyeG, eyeB shared lib
2. Then CNN kenerl and distortion the image then merge it by + operation
3. Send the image to dnn1 and State block
4. using the pre pixel data dnn1 decide the action (it reduce the calcualtion time as directly get from first state)
5. Send the dnn1 output data to the action shared lib.
6. Get reward by reward shared lib
7. Get new state again (we reusing the same propation, we just connect from +
   (don't worry about the overlapping it overwrite whenever start from the same start block)
8. Now by PPO add replay buffer and if the replay buffer is enough it calculate cost and update dnn1 and dnn2.

Shared library shape

Please see the src/extrafn.cpp file and save/state.json, act.json and rew.json

What is pre Input?

The preInput value is the initial value and if it dosen't have children, it keep that values until the program exit. Forexampel save/state.json has 7 pre vlauese, 1,2 are displace position 3,4 are capture display size, 5,6 are the resize for display, 7 is rgb channel option.

Q & A

Why the test is so slow?

It is not the algorithm problem the problem caused from extrafn shared library DLL file.

I wrote a very simple shared lib file for just testing the game, so don't use the test dll file for the real problems it is just for testing the program.

As referencing the form of the 'src/extrafn.cpp' resource file makes your own reward, action, and state functions.

As for giving a tip, most of taking time part is the OpenCV part and interface input part, if you directly get those data from your own program, the speed dramatically increases.

Only JSON file format allowed?

No!, I just wrote the file format as JSON, it is not required. you can change the extension name to anything but please keep the format.

Another tip Neural Network weight and bias value are based on base64 by Nyffenegger rene.nyffenegger@adp-gmbh.ch.

Why base64?

I have planned for multiplex socket TCP/IP code and also made it in the code but it is not used now. For that, It was preparation for networking.

Program unexpectedly shut down what can I do?

If you don't have an idea please open the terminal then type '*usg_AI --setting verboseon*'

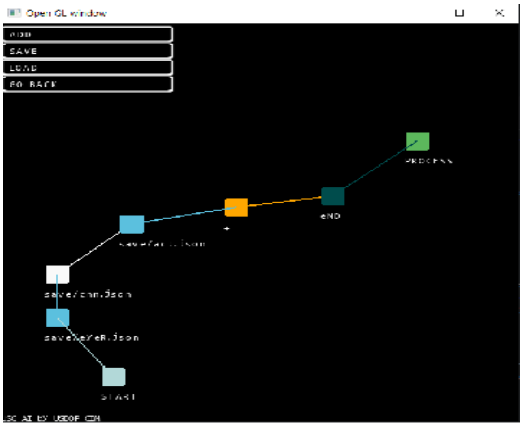
It shows every detail of the algorithm process.

On GUI, the block doesn't move and can't click!

Press Enter twice.

Can I use the graphical method for other programs?

Yes, you can make not only a Machine learning algorithm but also anything by the FUNC shared library block.



Only if you keep the format of it, the input and output working as the same as others.

Here is another example for making a button click macro by GUI *, not machine learning, just a macro program using OpenCV then moving keyboard step by step

```
1. 1
```