Intelligens Rendszerek

Öntanuló navigáció

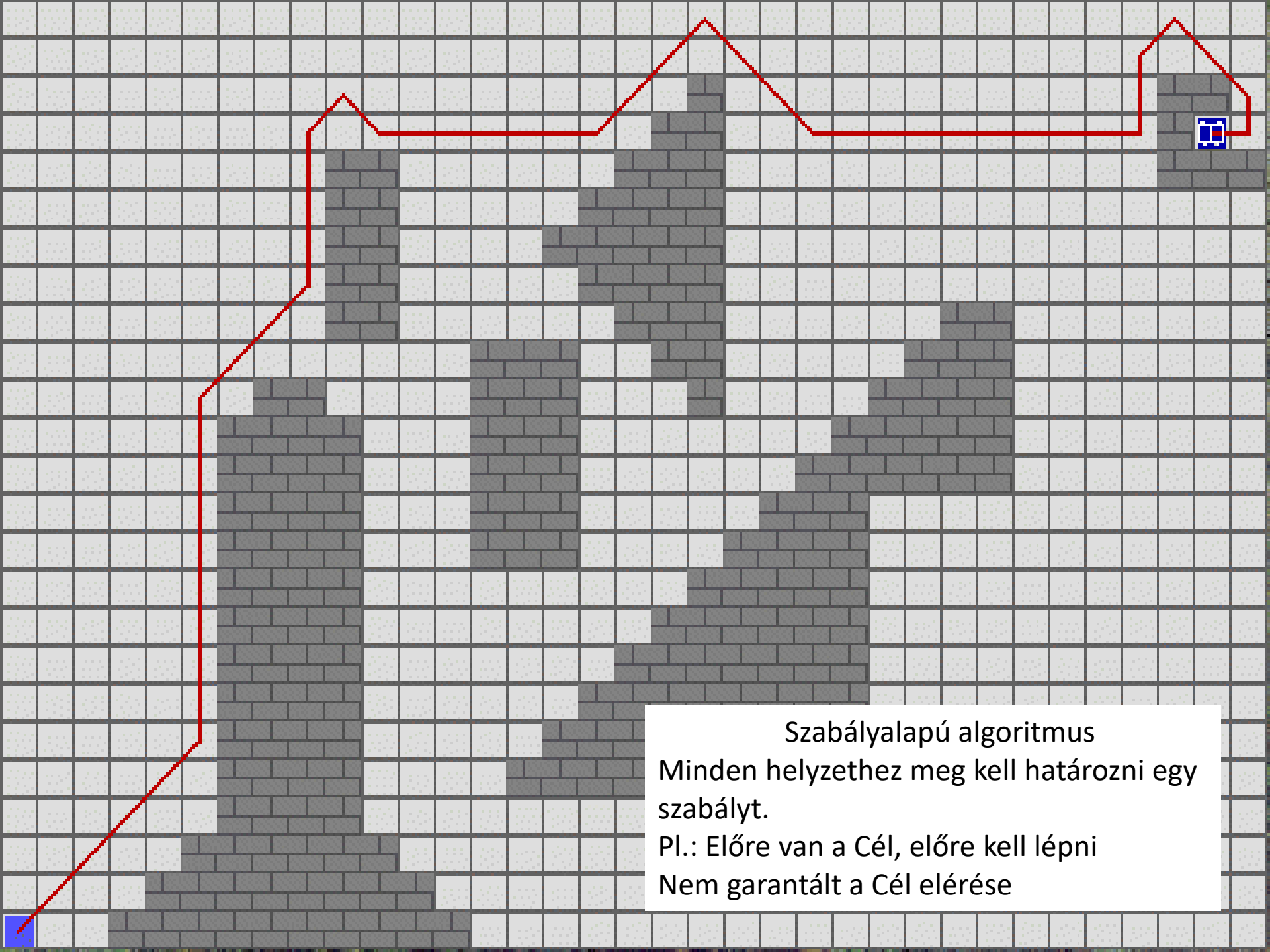
Navigáció

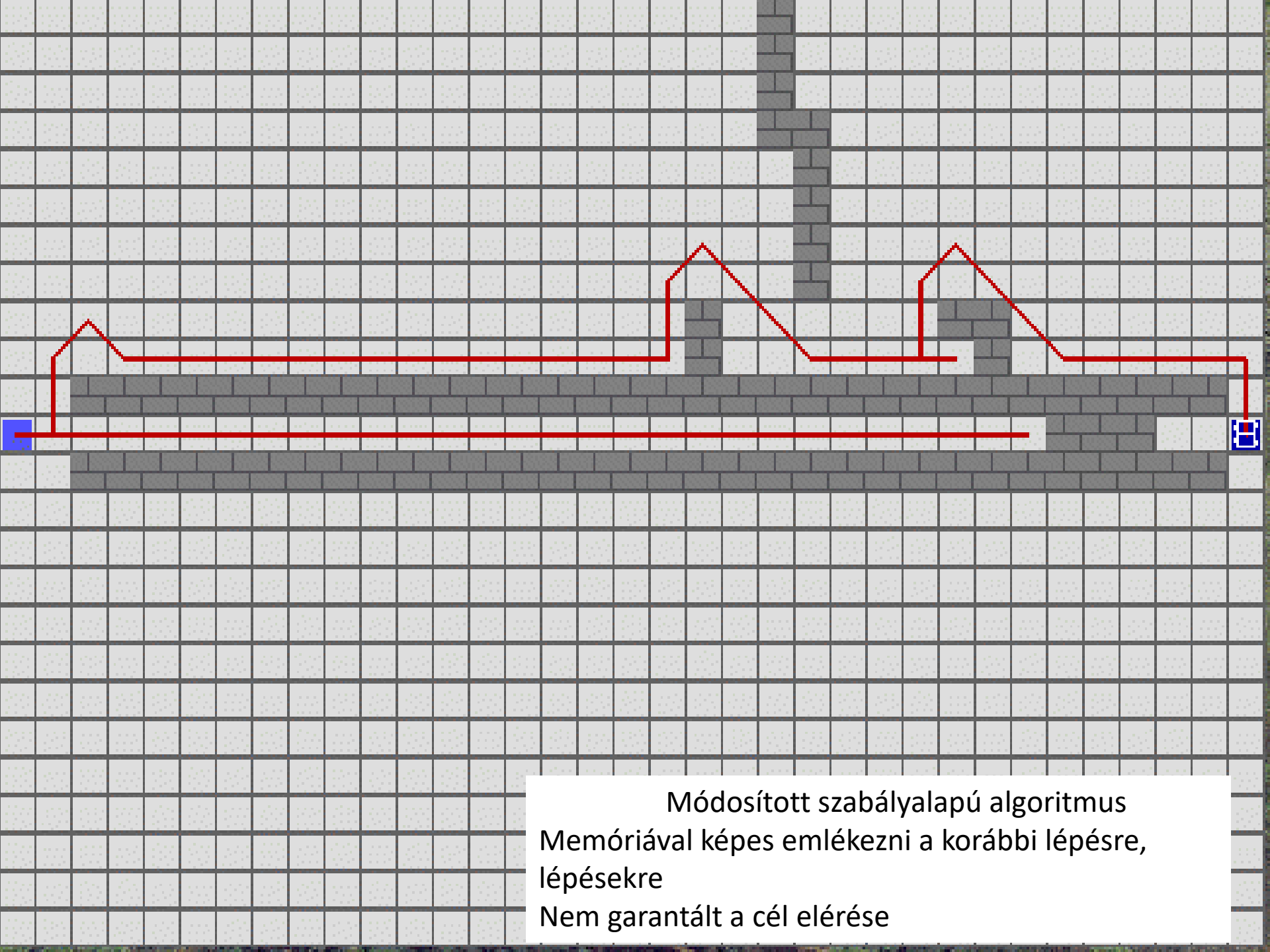
- A navigáció feladata, hogy a robot egy kijelölt célpozícióba jusson. A mozgás során több szempontot is figyelembe kell venni:
  - A robot mozgási lehetőségei
  - A mozgásra fordított energiaszükséglet
  - Idő
  - Robot mechanikai tulajdonságai
  - Terepviszonyok
- Ismert / ismeretlen terep

Navigáció

Akadályelkerülés és pályatervezés

- szabályalapú algoritmus
  - módosított szabályalapú algoritmus
  - neurális-elvű algoritmus
  - tapasztalat szerzésen alapuló algoritmus (öntanuló)
- hullám-továbbterjesztéses algoritmus
  - módosított hullám-továbbterjesztéses algoritmus
- GVD-elvű, gráfbejáráson alapuló algoritmus


Szabályalapú algoritmus
Minden helyzethez meg kell határozni egy szabályt.
Pl.: Előre van a Cél, előre kell lépni
Nem garantált a Cél elérése



Módosított szabályalapú algoritmus

Memóriával képes emlékezni a korábbi lépésre, lépésekre

Nem garantált a cél elérése

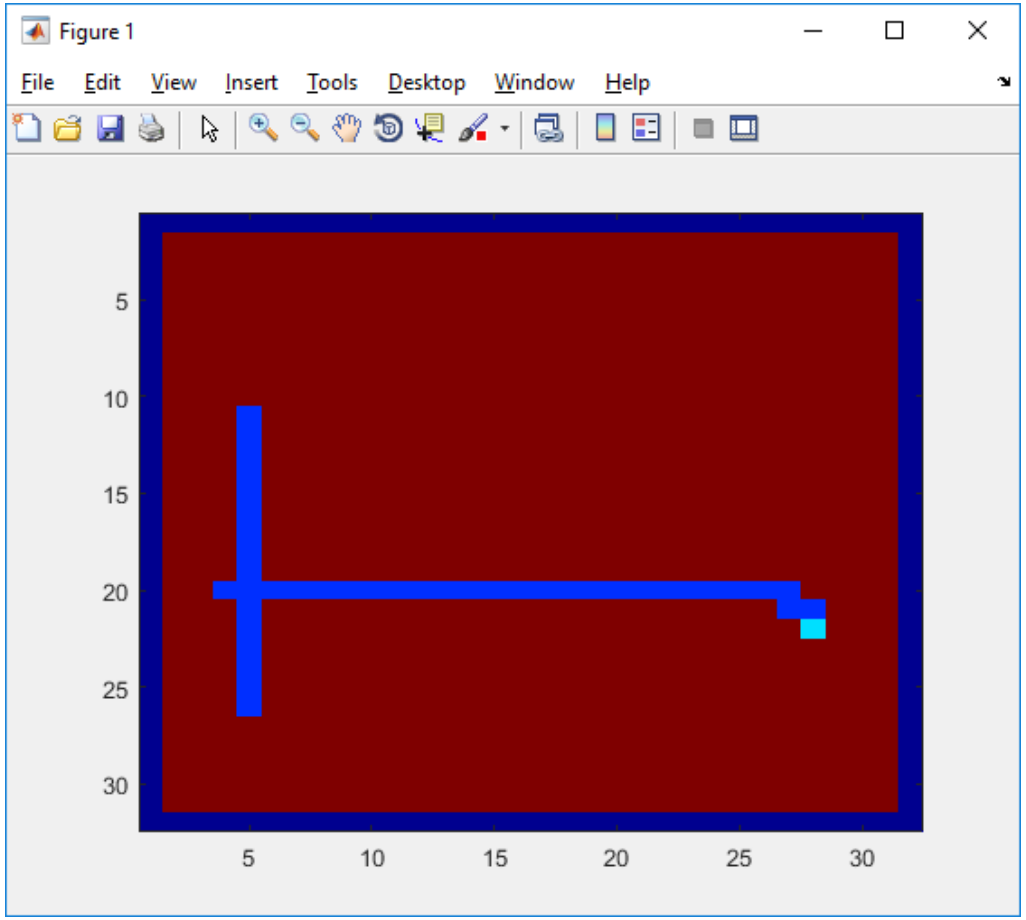
Öntanuló navigáció

- A szabálytábla kézi létrehozása és kitöltése helyett tapasztalati úton tanítjuk a rendszert.
- Meghatározzuk a robot kialakítása alapján a lehetséges lépéseket, majd a lekezelendő helyzeteket.
- Ez megad egy 2D szabálytáblát (min. 4x4 a mi esetünkben)
  - Jobbra, balra, fel, le irányban van a cél
  - Jobbra, balra, fel, le lépünk
  - Akadályt jelenleg nem kezelünk (üres térképen navigálunk)

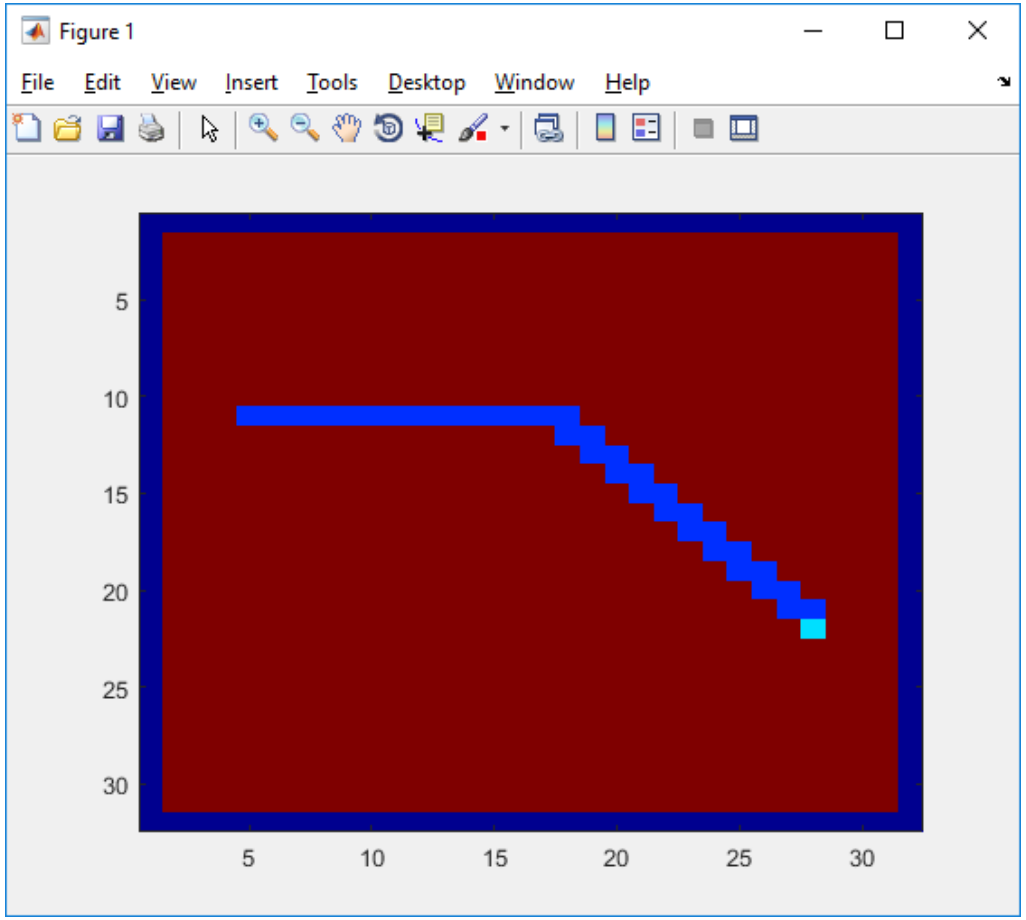
Öntanuló navigáció

- Navigáció lépései:
  - Kezdetben a szabálytábla üres
  - Irány meghatározása
    - Merre van a Cél mező a Robothoz képest?
    - Ez megadja a szabálytábla egy sorát
  - A szabálytáblában ki kell keresni az adott irányhoz tartozó legjobb lépést
    - A sorban ki kell választani a maximális értéket
    - Ennek index értéke mondja meg, hogy merre lépjen a robot (függetlenül attól, hogy „jó” vagy „rossz” irányba lépünk)
  - Lépés után meg kell vizsgálni, hogy közeledtünk, vagy távolodtunk a célhoz képet
    - Ha közeledtünk, az adott szabály jóság értékét növelni kell
    - Ha távolodtunk, az adott szabály jóság értékét csökkenteni kell
  - Új futtatásnál a szabálytábla tartalma (tapasztalat) megmarad

Öntanuló navigáció – kezdetben üres szabálytábla



Öntanuló navigáció – másodszori futtatás



Öntanuló navigáció

- Előnyök?
  - Ismert és ismeretlen terepen is működik
- Hátrányok?
  - Nem garantált a cél elérése
  - Nem tudni, hogy egyáltalán elérhető-e a cél

Matlab keretprogram

- Keretprogram indításkor rákérdez a szabálytábla törlésére (első indításnál mindig üres)
- Alaptérkép betöltése (32d.png)
  - Piros (255,0,0) cella: Robot
  - Zöld (0,255,0) cella: Cél
  - Fehér cellák (255,255,255): üres mező
  - Fekete cellák (0,0,0): akadály
- A keretprogram átalakítja ezt a képet (tömböt) egy 2D tömbbé, melyben a következő értékek szerepelnek:

```
emptyVal=0;
finishVal=-1;
robotVal=-2;
pathVal=-2.5;
obstacleVal=-3;
```

Matlab keretprogram

- Majd létrehozza az üres szabálytáblát (min. 4x4)

```
%szabálytábla sorai, oszlopai
directions=4;
movements=4;

%directions (irányok, merre van a cél?)
dirRight=1;
dirLeft=2;
dirUp=3;
dirDown=4;

%movements (lépések, merre léphetünk?)
moveRight=1;
moveLeft=2;
moveUp=3;
moveDown=4;
```

Matlab keretprogram

- Fő ciklus:

```
%% let's learn!
robotPos=startPos;

while(norm(robotPos)~=norm(finishPos))
     Addig fut, amíg a robot el nem ér a célba

     Q1: Mindig elér?
     Q2: Biztosítja valami azt, hogy ha nincs út
megáll?
```

1. feladat

directionIndex, azaz a robothoz képest a cél irányának meghatározása

2. feladat

movementIndex, azaz az adott irányhoz tartozó legnagyobb jóságú lépés kiválasztása a szabálytáblából

3. feladat

az új lépés kiszámítása, azaz "newPos" változó sorának ill. oszlopának inkrementálása ill. dekrementálása

4. feladat

távolság ellenőrzése, ha közeledtünk => szabály megerősítése, ha távolodtunk => szabály lerontása

A keretprogram végrehajtja a továbbiakat (léptetés, vizualizáció, stb.)