

Angewandte Informatik

Prof. Dr. Peter Gerwinski

10. Januar 2013

5.3 Wegfindungsalgorithmus für Roboterfahrzeug

5.3 Wegfindungsalgorithmus für Roboterfahrzeug

Projektaufgabe:

Ein RP6 wird in einer Ecke eines Labyrinths gestartet (dunkler Untergrund; helle Klebestreifen markieren Wände) und soll autark eine möglichst präzise Karte davon anfertigen.

Sensoren: Boden-Lichtsensoren, Kompaß

5.3 Wegfindungsalgorithmus für Roboterfahrzeug

Projektaufgabe:

Ein RP6 wird in einer Ecke eines Labyrinths gestartet (dunkler Untergrund; helle Klebestreifen markieren Wände) und soll autark eine möglichst präzise Karte davon anfertigen.

Sensoren: Boden-Lichtsensoren, Kompaß

Teilprobleme:

Wo war der Roboter schon?

Wo muß der Roboter noch suchen?

Wie kommt der Roboter möglichst effizient dorthin?

5.3 Wegfindungsalgorithmus für Roboterfahrzeug

Projektaufgabe:

Ein RP6 wird in einer Ecke eines Labyrinths gestartet (dunkler Untergrund; helle Klebestreifen markieren Wände) und soll autark eine möglichst präzise Karte davon anfertigen.

Sensoren: Boden-Lichtsensoren, Kompaß

Teilprobleme:

Wo war der Roboter schon?

Wo muß der Roboter noch suchen?

Wie kommt der Roboter möglichst effizient dorthin?

Vergleichbares Problem: Floodfill

Iterativer Floodfill-Algorithmus

Pseudo-Code: <http://de.wikipedia.org/wiki/Floodfill>

fill4 (x, y, alteFarbe, neueFarbe)

```
{
    stack.push (x, y);
    while (stackNotEmpty)
    {
        (x, y) = stack.pop;
        if (getPixel (x, y) == alteFarbe)
        {
            markierePixel (x, y, neueFarbe);
            stack.push (x, y + 1);
            stack.push (x, y - 1);
            stack.push (x + 1, y);
            stack.push (x - 1, y);
        }
    }
}
```

Iterativer Floodfill-Algorithmus

Pseudo-Code: <http://de.wikipedia.org/wiki/Floodfill>

fill4 (x, y, alteFarbe, neueFarbe)

```
{  
    stack.push (x, y);  
    while (stackNotEmpty)  
    {  
        (x, y) = stack.pop;  
        if (getPixel (x, y) == alteFarbe)  
        {  
            markierePixel (x, y, neueFarbe);  
            stack.push (x, y + 1);  
            stack.push (x, y - 1);  
            stack.push (x + 1, y);  
            stack.push (x - 1, y);  
        }  
    }  
}
```

Anstatt Selbstaufruf:
speichern, wo es weitergeht

Iterativer Floodfill-Algorithmus

Pseudo-Code: <http://de.wikipedia.org/wiki/Floodfill>

fill4 (x, y, alteFarbe, neueFarbe)

```
{
    stack.push (x, y);
    while (stackNotEmpty)
    {
        (x, y) = stack.pop;
        if (getPixel (x, y) == alteFarbe)
        {
            markierePixel (x, y, neueFarbe);
            stack.push (x, y + 1);
            stack.push (x, y - 1);
            stack.push (x + 1, y);
            stack.push (x - 1, y);
        }
    }
}
```

Wir benötigen einen Stack
für Koordinatenpaare.

→ Übung vom letzten Jahr

Anstatt Selbstaufruf:
speichern, wo es weitergeht

Iterativer Floodfill-Algorithmus

Pseudo-Code: <http://de.wikipedia.org/wiki/Floodfill>

fill4 (x, y, alteFarbe, neueFarbe)

```
{
    stack.push (x, y);
    while (stackNotEmpty)
    {
        (x, y) = stack.pop;
        if (getPixel (x, y) == alteFarbe)
        {
            markierePixel (x, y, neueFarbe);
            stack.push (x, y + 1);
            stack.push (x, y - 1);
            stack.push (x + 1, y);
            stack.push (x - 1, y);
        }
    }
}
```

Wir müssen uns merken,
wo wir schon waren.

Hier: Farbe
Roboter: notfalls Bit-Array

Anstatt Selbstaufruf:
speichern, wo es weitergeht

Iterativer Floodfill-Algorithmus

Pseudo-Code: <http://de.wikipedia.org/wiki/Floodfill>

fill4 (x, y, alteFarbe, neueFarbe)

```
{
    stack.push (x, y);
    while (stackNotEmpty)
    {
        (x, y) = stack.pop;
        if (getPixel (x, y) == alteFarbe)
        {
            markierePixel (x, y, neueFarbe);
            stack.push (x, y + 1);
            stack.push (x, y - 1);
            stack.push (x + 1, y);
            stack.push (x - 1, y);
        }
    }
}
```

Übungsaufgabe

Implementieren Sie einen iterativen Floodfill-Algorithmus für Text-Grafik.

In welcher Reihenfolge werden die Punkte besucht?

Iterativer Floodfill-Algorithmus

Pseudo-Code: <http://de.wikipedia.org/wiki/Floodfill>

fill4 (x, y, alteFarbe, neueFarbe)

```
{
    stack.push (x, y);
    while (stackNotEmpty)
    {
        (x, y) = stack.pop;
        if (getPixel (x, y) == alteFarbe)
        {
            markierePixel (x, y, neueFarbe);
            stack.push (x, y + 1);
            stack.push (x, y - 1);
            stack.push (x + 1, y);
            stack.push (x - 1, y);
        }
    }
}
```

Übungsaufgabe

Implementieren Sie einen iterativen Floodfill-Algorithmus für Text-Grafik.

In welcher Reihenfolge werden die Punkte besucht?

Ergebnis: In der falschen.

Iterativer Floodfill-Algorithmus

Pseudo-Code: <http://de.wikipedia.org/wiki/Floodfill>

fill4 (x, y, alteFarbe, neueFarbe)

```
{
    stack.push (x, y);
    while (stackNotEmpty)
    {
        (x, y) = stack.pop;
        if (getPixel (x, y) == alteFarbe)
        {
            markierePixel (x, y, neueFarbe);
            stack.push (x, y + 1);
            stack.push (x, y - 1);
            stack.push (x + 1, y);
            stack.push (x - 1, y);
        }
    }
}
```

Übungsaufgabe

Implementieren Sie einen iterativen Floodfill-Algorithmus für Text-Grafik.

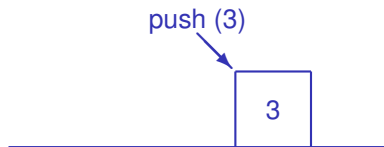
In welcher Reihenfolge werden die Punkte besucht?

Ergebnis: In der falschen.

Lösung: FIFO statt Stack

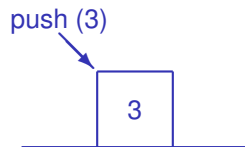
Was ist ein FIFO?

„First In – First Out“



FIFO = Queue = Reihe

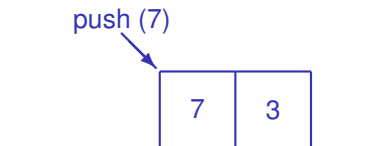
„Last In – First Out“



LIFO = Stack = Stapel

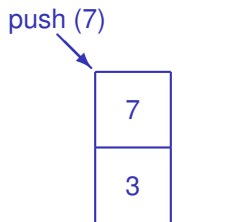
Was ist ein FIFO?

„First In – First Out“



FIFO = Queue = Reihe

„Last In – First Out“



LIFO = Stack = Stapel

Was ist ein FIFO?

„First In – First Out“

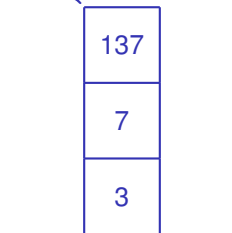
push (137)



FIFO = Queue = Reihe

„Last In – First Out“

push (137)



LIFO = Stack = Stapel

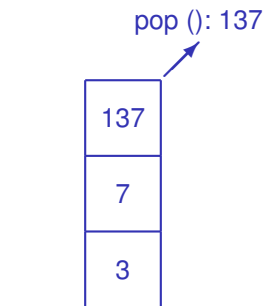
Was ist ein FIFO?

„First In – First Out“



FIFO = Queue = Reihe

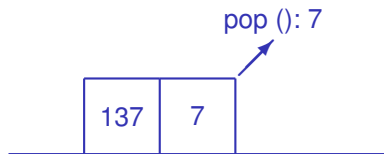
„Last In – First Out“



LIFO = Stack = Stapel

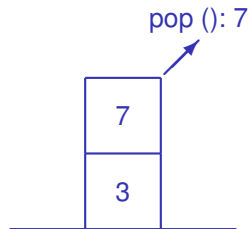
Was ist ein FIFO?

„First In – First Out“



FIFO = Queue = Reihe

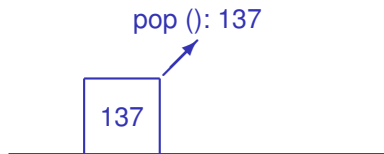
„Last In – First Out“



LIFO = Stack = Stapel

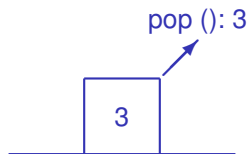
Was ist ein FIFO?

„First In – First Out“



FIFO = Queue = Reihe

„Last In – First Out“



LIFO = Stack = Stapel

Übungsaufgabe: FIFO

Implementieren Sie einen FIFO für Koordinaten-Paare.
Koordinatenpaare sollen „hintereinander eingereiht“ werden.

- Eine Funktion `push (int x, int y)`
schiebt sie in den FIFO.
- Eine Funktion `pop (int &x, int &y)`
liest und entfernt sie wieder aus dem FIFO.

Übungsaufgabe: FIFO

Implementieren Sie einen FIFO für Koordinaten-Paare.
Koordinatenpaare sollen „hintereinander eingereiht“ werden.

- Eine Funktion `push (int x, int y)`
schiebt sie in den FIFO.
- Eine Funktion `pop (int &x, int &y)`
liest und entfernt sie wieder aus dem FIFO.

Anleitung:

- 1 Array
- 2 Index-Variablen (eine zum Schreiben, eine zum Lesen)
- ansonsten wie Stack

Iterativer Floodfill-Algorithmus mit Stack

Pseudo-Code:

```
fill4 (x, y, alteFarbe, neueFarbe)
{
    stack.push (x, y);
    while (stackNotEmpty)
    {
        (x, y) = stack.pop;
        if (getPixel (x, y) == alteFarbe)
        {
            markierePixel (x, y, neueFarbe);
            stack.push (x, y + 1);
            stack.push (x, y - 1);
            stack.push (x + 1, y);
            stack.push (x - 1, y);
        }
    }
}
```

Iterativer Floodfill-Algorithmus mit FIFO

Pseudo-Code:

```
fill4 (x, y, alteFarbe, neueFarbe)
{
    fifo.push (x, y);
    while (fifoNotEmpty)
    {
        (x, y) = fifo.pop;
        if (getPixel (x, y) == alteFarbe)
        {
            markierePixel (x, y, neueFarbe);
            fifo.push (x, y + 1);
            fifo.push (x, y - 1);
            fifo.push (x + 1, y);
            fifo.push (x - 1, y);
        }
    }
}
```

Iterativer Floodfill-Algorithmus mit FIFO

Pseudo-Code:

```
fill4 (x, y, alteFarbe, neueFarbe)
{
    fifo.push (x, y);
    while (fifoNotEmpty)
    {
        (x, y) = fifo.pop;
        if (getPixel (x, y) == alteFarbe)
        {
            markierePixel (x, y, neueFarbe);
            fifo.push (x, y + 1);
            fifo.push (x, y - 1);
            fifo.push (x + 1, y);
            fifo.push (x - 1, y);
        }
    }
}
```

Übungsaufgabe

Implementieren Sie einen iterativen Floodfill-Algorithmus mit FIFO für Text-Grafik.

In welcher Reihenfolge werden die Punkte besucht?

Iterativer Floodfill-Algorithmus mit FIFO

Pseudo-Code:

```
fill4 (x, y, alteFarbe, neueFarbe)
{
    fifo.push (x, y);
    while (fifoNotEmpty)
    {
        (x, y) = fifo.pop;
        if (getPixel (x, y) == alteFarbe)
        {
            markierePixel (x, y, neueFarbe);
            fifo.push (x, y + 1);
            fifo.push (x, y - 1);
            fifo.push (x + 1, y);
            fifo.push (x - 1, y);
        }
    }
}
```

Übungsaufgabe

Implementieren Sie einen iterativen Floodfill-Algorithmus mit FIFO für Text-Grafik.

In welcher Reihenfolge werden die Punkte besucht?

Ergebnis: In der richtigen. :-)