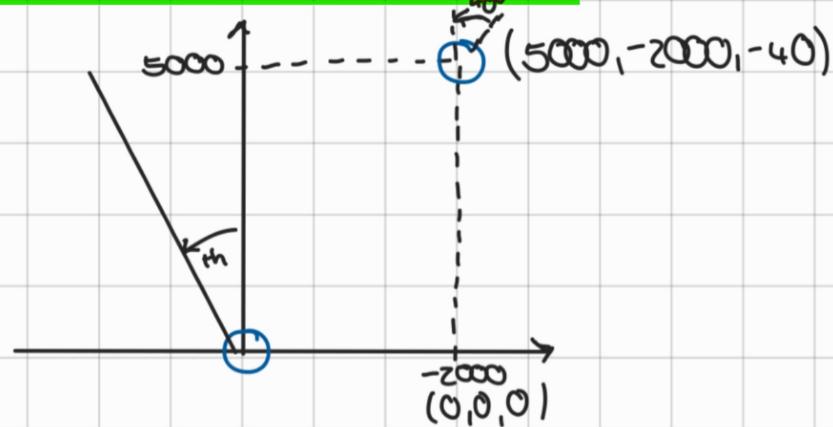


Vereinfachung bei ENU in einer Ebene

8. VE

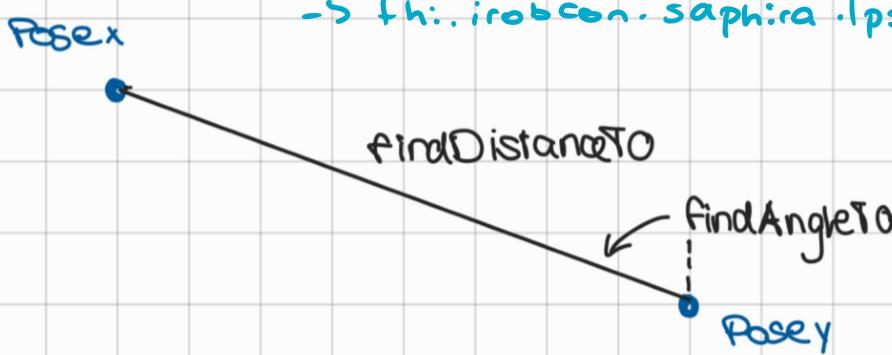


Auslesen Roboterposition mit `getPose()` als Pose:

- * `get()`, `set()` für x, y, θ_h
- * `double x = findDistanceTo(Pose y)`

Einfacher Abstand zwischen x und y

$\rightarrow \text{th} \ldots \text{irobotcon.saphira.lps} (\text{Pose.java})$



- * `double x.findAngleTo(Pose y)`
Einfacher Winkel zwischen x und y
- * `double x.diffAngle(Pose y)`
Differenzwinkel Orientierungen von x und y



$\odot y$ diffAngle

5.4.2

Bisher: Abfrage Sonare einzeln und direkt

Probleme:

- * Wiederverwendung Programme auf untersch. Robotern
- * Entfernungsmessung nicht nur mit Sonaren

Abhilfe:

- a) Konfiguration Multisensor

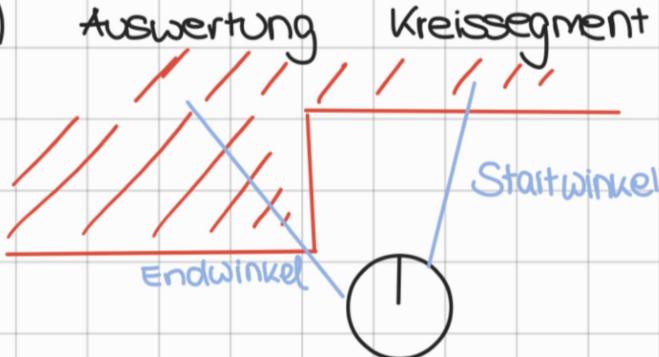
- b) Abfrage fusionierter Sensordaten in Regions of Interests

In THIRobot:

a) robot.addLaser(1) // Einbindung Laserscanner in LPS
 robot.addVCC4() // Einbindung Kamera

b)

b1) Auswertung Kreissegment



Methode:

double checkPolar()

Beispiel: Orthogonales Andocken

Abstand zum
Objekt

// Mit Kegel

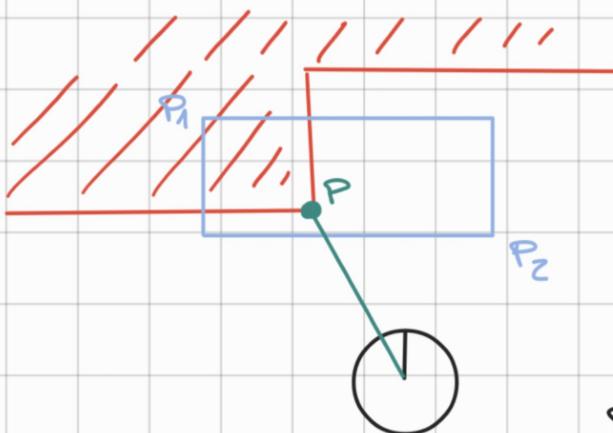
leftDist = (int) checkPolar(
 Start, End);

hypotest Java

(robot.checkPolar(2, 10, relObstaclePose) - robot.getRadius());

rightDist = (int)(checkPolar(-10, -2, null) - robot.getRadius());

Abstand
bereagen
auf Gehäuse



Methode:

checkBox()

Beispiel: Orthogonales Andocken

// mit Box

leftDist = (int) (robot.checkBox(2500, 500, 100, 100, relObstaclePose)
 - robot.getRadius());

rightDist = (int) (robot.checkBox(2500, -500, 100, -100, null)
 - robot.getRadius());

Eintragen in globale Karte erfordert Koordinatentransformation
 aus Roboterkoordinatensystem in Basiskoordinatensystem

Beispiel: 2D - Koordinatentransformation

Drehmatrix & Verschiebungsvektor

$$\begin{pmatrix} \cos \alpha & -\sin \alpha \\ \sin \alpha & \cos \alpha \end{pmatrix}$$

$$\begin{pmatrix} d_x \\ d_z \end{pmatrix}$$

Podsacke
relative
Hindernisposition
(1000, 1800, 0)

1800
1000

Robot position

Ursprung
Roboterkoordinatensystem

2000

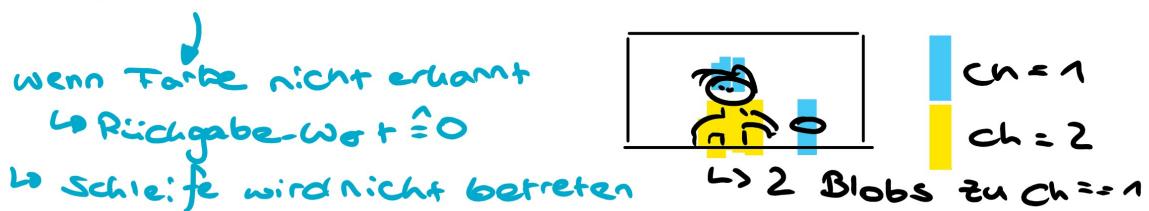
Ursprung
Weltkoordinatensystem

45°

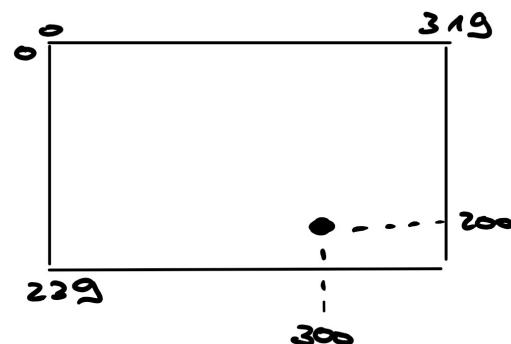
$$P_{\text{obst}} = \begin{pmatrix} \cos 45 & -\sin 45 \\ \sin 45 & \cos 45 \end{pmatrix} \cdot \begin{pmatrix} 1000 \\ 1800 \end{pmatrix} + \begin{pmatrix} 3000 \\ 2000 \end{pmatrix} = \begin{pmatrix} 4950 \\ 3960 \end{pmatrix}$$

5.4.3 Bilder

- Merkmalserkennung über Farben und Kanten
 - Farbbereiche durch Region Growing in Echtzeit
↳ ACTS
 - Kantenerkennung durch Hough Transformation
↳ nicht in Echtzeit (Sekundenbereich)
↳ HALCON
 - Trainieren Farbbereiche offline zu Kanälen
 - Auswertung in THIRobCon
 - int getNumBlobs (int ch) Anzahl Blobs im Kanal
 - wenn Farbe nicht erkannt
↳ Rückgabe-Wert = 0
 - ↳ Schleife wird nicht betreten
- Blob getBlob (int ch, int i) Auslesen Blob i zum Kanal ch in Schleife

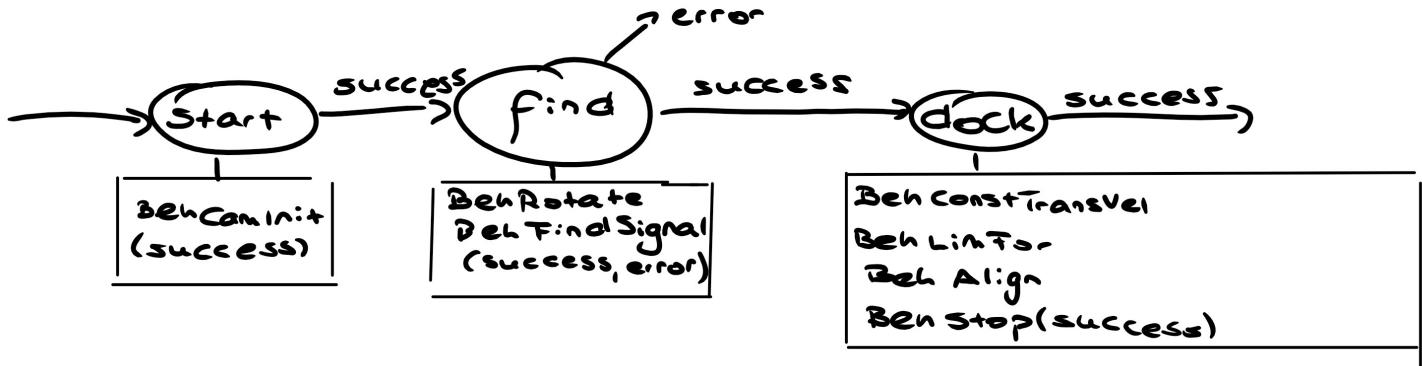


- Auswertung Blob:
 - int getXCG() CG-Center of Gravity
 - int getYCG() Koordinaten Blobschwerpunkt bezogen auf Bildgröße
- int getArea() Zahl Pixel im Blob
- int getTop(), ... , getRight() Ausdehnung Blob



- Positionierung der Kamera:
 - BehCam Init - Startausrichtung der Kamera
yaw → pan und tilt ← pitch
 - DesCam Pan, DesCamTilt dynamisch im Verhaltensmuster (Kamera bewegen/drehen etc.)

Beispiel: SignalDock



```
1 package thi.irobcon.app.behaviour.signaldock;
2
3 import thi.irobcon.saphira.proactive.Strategy;
4
5 public class SDStrategy extends Strategy {
6
7     protected BehGroup start, find, dock;
8
9     public SDStrategy(BehGroup start, BehGroup find, BehGroup dock) {
10        super();
11        this.start = start;
12        this.find = find;
13        this.dock = dock;
14        addOutput("Try to start ...");
15        start.activateExclusive();
16    }
17
18    public void plan() {
19        if (start.isSuccess()) {
20            addOutput("done\nTry to find ...");
21            find.activateExclusive();
22        }
23        if (find.isError()) {
24            addOutput("done\nNot found ... stop");
25            stopRunning();
26        }
27        if (find.isSuccess()) {
28            addOutput("done\nTry to dock ...");
29            dock.activateExclusive();
30        }
31        if (dock.isSuccess()) {
32            addOutput("done\nSucceeded ... stop");
33            stopRunning();
34        }
35    }
36}
```

```
1 package thi.irobcon.app.behaviour.signdock;
2
3•import thi.irobcon.saphira.desire.DesRotVel;□
4
5 public class BehRotate extends Behaviour {
6
7     protected int rotVel;
8
9
0•public BehRotate(String actionPerformed, int rotVel) {
1     super(actionName);
2     this.rotVel = rotVel;
3 }
4
5•public void fire() {
6     addDesire(new DesRotVel(rotVel, 1.0));
7 }
8 }
9
0
```

```
1 package thi.irobcon.app.behaviour.signdock;
2
3•import thi.irobcon.saphira.lps.Blob;□
4
5 public class BehFindSignal extends Behaviour {
6
7     protected int channel, timeout;
8
9
0•public BehFindSignal(String actionPerformed, int channel, int timeout) {
1     super(actionName);
2     this.channel = channel;
3     this.timeout = timeout;
4 }
5
6•public void fire() {
7     if (timeout == 0) error();
8     else timeout--;
9
0     int numBlobs = robot.getNumBlobs(channel);
1     System.out.println("NumBlobs: " + numBlobs);
2
3     if (timeout == 10) success();
4
5     if (numBlobs > 0) {
6         Blob blob = robot.getBlob(channel, 0);
7         if (blob.getXCG() > 70 && blob.getXCG() < 90)
8             success();
9     }
0 }
1 }
2
```

```

1 package thi.irobcon.app.behaviour.signdock;
2
3 import thi.irobcon.app.behaviour.rightangledock.BehAlign;□
4
5 public class SDMain {
6
7     public static void main(java.lang.String[] args) {
8
9         // SaphiraRobot robot = new SaphiraRobot(ECarDefines.DX3);
10        SaphiraRobot robot = new SaphiraRobot(ECarDefines.JSIM_DX3);
11
12        // robot.addLaser();
13        robot.addHalcon("");
14        robot.addPTZCamWithACTS("/robotics/...actsconfig", "1");
15
16        // start
17        BehGroup start = new BehGroup("Start");
18        BehCamInit ci = new BehCamInit("CamInit", 0, -10, 0);
19        start.add(ci, 50); → Wahlfrei
20        robot.add(start);
21
22        // find
23        BehGroup find = new BehGroup("Find");
24        BehRotate cr = new BehRotate("Rotate", 20);
25        BehFindSignal fs = new BehFindSignal("FindSignal", 1, 50);
26        find.add(cr, 50);
27        find.add(fs, 50);
28        robot.add(find);
29
30
31        // dock
32        BehGroup dock = new BehGroup("Dock");
33        BehConstTransVel cv = new BehConstTransVel("ConstVel", 400);
34        BehLimFor lf = new BehLimFor("LimFor", 500, 2000, 100);
35        BehAlign al = new BehAlign("Align", 30, 5);
36        BehStop st = new BehStop("Stop", 800);
37        dock.add(lf, 80);
38        dock.add(cv, 50);
39        dock.add(al, 75);
40        dock.add(st, 80);
41        robot.add(dock);
42
43        // strategy
44        robot.add(new SDStrategy(start, find, dock));
45
46        robot.run();
47    }
48}

```