TP3 : Comportement temps réel de l'agent Perception

S.Benaissa
GI3-ENSEM-CASA
said.benaissa@ensem.ac.ma

2015–2016

1 Introduction

Un robot est une machine équipée de capacités de perception, de décision et d'action qui lui permettent d'agir de manière autonome dans son environnement en fonction de la perception qu'il en a.

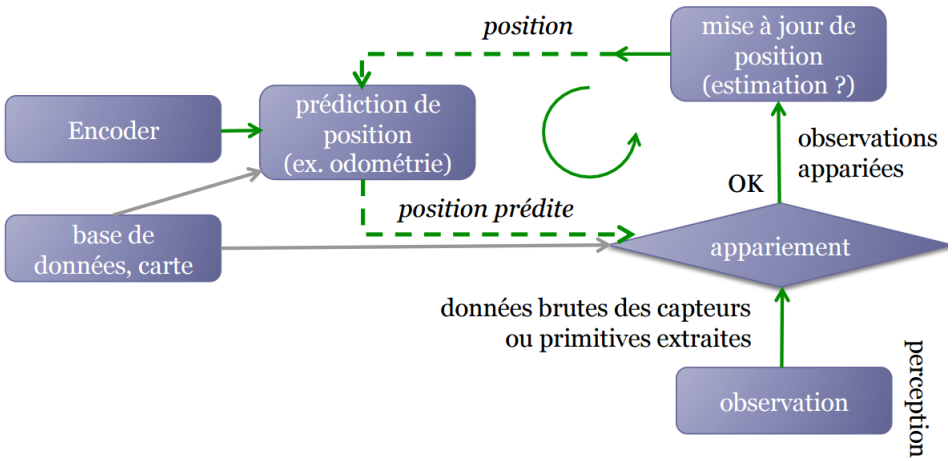

FIGURE 1 – Architecture d'agent localisation de robot mobile.

2 Modèle d'agent de perception

$R(O, x, y)$ le repère fixe lié à l'environnement externe. La position de robot en k^{em} itération est $P_k = [x_k, y_k, z_k]$ avec x_k et y_k les coordonnées 2D et θ_k

l'orientation dans $R(O, x, y)$.

$R^{'}(O^{'}, x^{'}, y^{'})$ le repère mobile lié au robot mobile khepera-III avec $O^{'}$ le centre de masse de robot. Soit $P_k^{'} = [x_k^{'}, y_k^{'}, z_k^{'}]$ représente le déplacement et l'orientation par rapport a $R^{'}(O^{'}, x^{'}, y^{'})$.
Dans le repère d'environnement $R(O, x, y)$ le déplacement relatif Δd_k et l'orientation $\Delta\theta_k$ de robot entre deux itérations élémentaires sont :

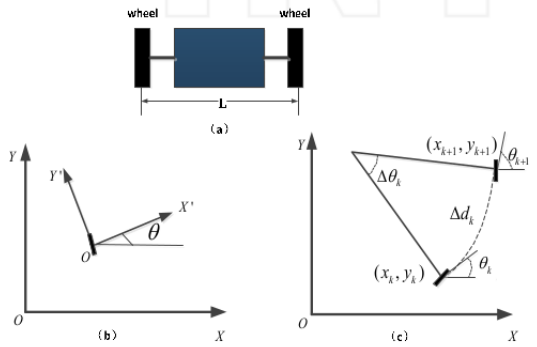

FIGURE 2 – (a) :Robot mobile khepera IV. (b) : Système de coordonnée $R(O, x, y)$ et $R^{'}(O^{'}, x^{'}, y^{'})$. (c) :Modèle de localisation de robot mobile.

$$\begin{cases} \Delta d_k = \frac{\Delta d_{k,r}^{'} + \Delta d_{k,l}^{'}}{2} \\ \Delta\theta_k = \frac{\Delta d_{k,r}^{'} - \Delta d_{k,l}^{'}}{L} \end{cases}$$

$\Delta d_{k,r}^{'}$ et $\Delta d_{k,l}^{'}$ représente les déplacements relatifs respectivement des roues droit et gauche. ces deux valeurs seront mesuré par les capteurs odometriques installés sur chacune des deux roues. $L = 140mm$ est la distance entre les centres des deux roues de robot. $k = 1, 2, 3, ..., n$ représentant l'itération de mesure. Pour chaque mesure k on a la position de robot :

$$\begin{cases} x_{k+1} = x_k + \Delta d_k \cos(\theta_k + \frac{\Delta\theta_k}{2}) \\ y_{k+1} = y_k + \Delta d_k \sin(\theta_k + \frac{\Delta\theta_k}{2}) \\ \theta_{k+1} = \theta_k + \Delta\theta_k \end{cases}$$

3 A.Perception : Position Prédite

```
function [x, y, theta] = MajOdom(xk, yk, thetak, \delta-dg, delta_dd, L)
delta_d = (delta_dg+delta_dd)/2;   // Calcule de déplacement de centre de robot.
delta_theta = (delta_dd-delta_dg)/L; // Calcule d'orientation de centre de robot.
x = xk + delta_s*cos(thetak + delta_theta/2); // Mise à jour sur x
y = yk + delta_s*sin(thetak + delta_theta/2); // Mise à jour sur y
theta = thetak + delta_theta;                 // Mise à jour sur l'orientation
```

- Programmer l'algorithme de mise à jour de position de robot.
- Alimenter l'algorithme par des valeurs initiales.
- Analyser le comportement temps réel d'agent Perception (Cas Position Prédiction).

4 A.Perception : Position Mesurée

1. Utiliser le code "rodo.c" pour mesures odométriques.
2. Compiler "rodo.c" via la commande "make".
3. Utiliser la commande ssh root@192.168.1.2 './rodo' > data.txt pour récupérer les mesures vers un fichier "data.txt" sur votre machine.
4. Alimenter l'algorithme par des valeurs mesurées.
5. Analyser le comportement temps réel d'agent Perception (Cas Position Estimation odométrique).

5 Perception : Aspect T.Réel

Analyser le comportement temps-réel d'agent perception dans le cas des mesures réel(estimation odométriques) et dans le cas de prédiction.

1. La distance entre les centres des deux roues de robot : L=140mm.
2. Relation entre la vitesse en pulsation vers la vitesse en mm/s : 1 pulse/10ms = 0.678 mm/s.
3. Relation entre la position absolue en pulsation vers la position absolue en mm : 1 pulse = 0.006781 mm.

6 Code source

Listing 1 – Code source pour mesures odométriques.

```
/*S.Benaissa (said.benaissa@ensem.ac.ma)
 * compiler via la commande: make
*/
#include <khepera/khepera.h>
#include <signal.h>
static knet_dev_t * dsPic;
int maxsp,accinc,accdiv,minspacc,minspdec;
static int quitReq=0;
int main(int argc,int *argv[])
{
int i,lpos,rpos;
long motspeed=200;
printf("\n Read odom \n");
```

```
// init libkhepera
if(kh4_init(0,NULL)!=0)
{
printf("\n init de libkhepera!\n\n");
return -1;
}
dsPic=knet_open("Khepera4:dsPic",KNET_BUS_I2C,0,NULL);
if(dsPic==NULL)
{
printf("\n prob comm Kh4 dsPic\n\n");
return -2;
}
while(100)
{
kh4_SetMode( kh4RegSpeedProfile,dsPic );
kh4_set_speed(motspeed,motspeed,dsPic);
kh4_get_position(&lpos,&rpos,dsPic);
printf("\n odom: left %ld | right %ld\n",lpos,rpos);
}
kh4_SetMode(kh4RegSpeedProfile,dsPic);
kh4_set_speed(0,0,dsPic);
}
```

Listing 2 – Makefile de compilation.

```
# S.Benaissa (said.benaissa@ensem.ac.ma)
#  Makefile for development with libkhepera
.PHONY: clean
# Modify this!
#KTEAM_HOME = ../..
KTEAM_HOME =~/khepera4_development
# And maybe these
LIBKHEPERA_ROOT  = ${KTEAM_HOME}/libkhepera-1.0
TARGET_SYSTEM    = khepera-2.6
OE_HOME = /usr/local/khepera4-oetools
KHEPERA_TOOLS = ${OE_HOME}
KTEAM_KERNEL_VERSION = 2.6.24
KTEAM_KERNEL_HOME = ${KHEPERA_TOOLS}/tmp/work/
overo-angstrom-linux-gnueabi/linux-omap3-${KTEAM_KERNEL_VERSION}-r97/git
# And don't touch these
ARCH = arm
CROSS_COMPILE = arm-angstrom-linux-gnueabi-
PATH := $(PATH):${KHEPERA_TOOLS}/tmp/sysroots/i686-linux/usr/armv7a/bin
CC = ${CROSS_COMPILE}gcc
CXX = ${CROSS_COMPILE}g++
LD = ${CROSS_COMPILE}ld
AR = ${CROSS_COMPILE}ar
```

```
AS = ${CROSS_COMPILE}as
INCPATH = ${LIBKHEPERA_ROOT}/build-${TARGET_SYSTEM}/include
LIBPATH = ${LIBKHEPERA_ROOT}/build-${TARGET_SYSTEM}/lib
# Pointer to the libkhepera build directory
LIBKHEPERA = ${LIBKHEPERA_ROOT}/build-${TARGET_SYSTEM}
SRCS    = $(wildcard *.c)
OBJS    = $(patsubst %.c,%.o,${SRCS})
INCS    = -I ${LIBKHEPERA}/include
LIBS    = -L ${LIBKHEPERA}/lib -lkhepera -lpthread
CFLAGS  = -O2
# for debugging
#CFLAGS         = -g
TARGET  = template template-static
.PHONY: all clean depend
r_odo: r_odo.o
        @echo "Building $@"
        $(CC) -o $@ $? $(LIBS) $(INCS) $(CFLAGS)
template-static: r_odo.o
        @echo "Building $@"
        @$(CC) -o $@ $? $(LIBS) -static $(INCS) $(CFLAGS)
all:    ${TARGET}
clean :
        @echo "Cleaning"
        @rm -f *.o .depend ${TARGET} *~
depend:
        @echo "Building dependencies"
        @rm -f .depend
        @touch .depend
        @makedepend ${SYS_INCLUDES} ${INCS} -Y -f .depend ${SRCS}
%.o:    %.c
        @echo "Compiling $@"
        $(CC) $(INCS) -c $(CFLAGS) $< -o $@
ifeq (.depend,$(wildcard .depend))
include .depend
endif
```