Robot localization based on KS-FAM

July 29, 2010

1 Description

The objective is mobile robot vision based localization using associative memories. The map stores a path previously followed by the robot in the form of several view “landmarks” representing points of interest in the path. Those landmarks will identify a section of the path, dividing it in a sequence of locations without gaps between them. These landmarks are stored as gray-scale patterns in a KS-FAM. Localization will be performed by feeding the KS-FAM with the images that the robot acquires in its movement, obtaining from it the recognized position.

2 Experiment details

For the experiment, the optical image database already recorded is used.

The code for the KS-FAM was provided by prof. Peter Sussner[1].

Available example uses of KS-FAM are as Auto-Associative memories. In this experiment, the Auto-Associative type has the additional problem of estimating which position is the one recalled by the memory. Visual examination of results with both Auto-Associative and Hetero-Associative memories seemed to give very similar results. So, in a first approach, Hetero-Associative memories are used.

In the pairs (x,y), x will be the pattern (gray-scale image corresponding to the landmark that is going to be stored) and y will be a vector of size n $=$ # of patterns to store. The vector will be composed of 0’s, except for one 1 in the vector position corresponding to the map position of the stored pattern. e.g:

Being $X = \{x_1, x_2, x_3, x_4, x_5\}$ the patterns that we want to encode in the KS-FAM. The pair y_2 of pattern x_2 (second pattern in the path) will be $y_2 = [01000]$. Y (the matrix of outputs) will be then (vectors stored column-wise):

[1] http://www.ehu.es/ccwintco/groupware/webdav.php/apps/phpbrain/142/KSFAM%20-%20Code.rar

$$Y = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix},$$

which corresponds to an identity matrix of size nxn.

Initially, a simpler approach was used, being y_i a scalar identifying the position (i.e. '2' for the second position instead of [01000]). However, results obtained with that method were much worse.

For validation purposes, the same ground division based on the odometry data of previous experiments has been used.

3 Implementation details

First, the image database is transformed to gray-scale [0,1], as is done in the sample code provided by Sussner.

```
for i = 1:nWalks
        for j = 1:tamsBD(i);
                bdImagenes{i}(j,:) = mat2gray(bdImagenes{i}(j,:));
        end
end
```

The patterns matrix is built using the images of the selected landmark positions from the first walk.

```
X = zeros(tamVec, nSitios); % reservo espacio para matriz de patrones
% obtengo los patrones (imagenes de los landmaks)
for i = 1:nSitios
        X(:,i) = bdImagenes{1}(sitios(i), :);
end
```

Output patterns matrix is built as the identity matrix.

```
Y = eye(nSitios); % cada vector tendrá un 1 en la posición correspondiente
```

Mxz and Wzy memories are built using the input and output pattern matrices.

```
Mxz = BoxMax2(eye(nSitios), -1*X',-Inf);
Wzy = BoxMin2(Y, -1*eye(nSitios), Inf);
```

For each test walk i, the images are put in an input matrix and feed to the memories. Some of the code is redundant or unnecessary, but was done like that to make sure that it was being done correctly.

```
Xin = zeros(tamVec, tamsBD(i));
for j = 1:tamsBD(i)
        Xin(:,j) = bdImagenes{i}(j,:);
end
[Yout,u]  =  AMM_Nova(Xin,Mxz,Wzy);
```

Output vectors are translated to scalars identifying the positions ('find' returns the nonzero position in the vector) .

```
posLoc(j) = find(Yout(:,j));
```

Success rate is calculated for each walk ($i+1$ because the first walk was used for training) using the path division based on odometry.

```
aciertos(i) = sum(posLoc{i}(:) == gruposOdo{i+1}(:))/tamsBD(i+1);
```

4 Results

Obtained results are rather poor, as can be appreciated in table 1. Surprisingly, the best results were obtained using the smallest images.

Image size	**Walk 2**	**Walk 3**	**Walk 4**	**Walk 5**	**Walk 6**	**Mean**
242x314	0.3221	0.3812	0.2883	0.3264	0.246	0.3128
121x157	0.2969	0.3193	0.2909	0.3107	0.2086	0.28528
61x79	0.4678	0.4629	0.4494	0.389	0.4171	0.43724

Table 1: Position recognition success rates obtained using images of different sizes.