Um Arquivo Exemplo em Latex

G. A. Borges

March 31, 2005

1 Introduction

Environment maps are extensively used in mobile robots navigation systems for tasks like map-based positioning or global path planning. When the robot moves on a flat ground, a 2-D map is sufficient to solve these problems. For indoor environments, a such map can be extracted from a set of range data provided by a rotating laser range finder. Planar surfaces that often occur in structured environments are modeled by line segments. The process of line extraction from range data must be executed on line while the robot is moving, and it must provide an accurate polygonal model of the environment. Classical algorithms initially used for this task [**?**, **?**, **?**] are issued from the edge segmentation methods developed for video image processing. These algorithms are generally very sensitive to changes on some parameters. More sophisticated methods like prototype-based fuzzy clustering algorithms [**?**, **?**] are more robust but they are generally more time expensive. In order to avoid some drawbacks of these solutions, this paper proposes a split-and-merge segmentation algorithm based on the fuzzy clustering approach.

In this paper, Section 2 discuss about fuzzy clustering.

2 Prototype-based fuzzy clustering algorithm

The main objective of prototype-based fuzzy clustering algorithms is to reduce iteratively a cost function J. Most algorithms use a cost function given by

$$J(\beta,\mathcal{U},\mathcal{Z}) = \sum_{i=1}^{C}\sum_{j=1}^{N} u_{ij}^{m} d^2(x_j, y_j, \beta_i). \tag{1}$$

In eq. (1), β_i represents the parameters of the *i-th* prototype and u_{ij}^m is the grade of membership of the *j-th* point (x_j, y_j) to the prototype β_i. $d(x_j, y_j, \beta_i)$ is a distance function between the point (x_j, y_j) and the prototype β_i and m is a constant. The fuzzy clustering algorithm also imposes the following constraints:

$$u_{ij} \in [0,1], \quad 0 < \sum_{j=1}^{N} u_{ij} < N, \tag{2}$$

$$\sum_{i=1}^{C} u_{ij} = 1, \tag{3}$$

For the purposes of straight lines extraction, there exist different representations for the prototypes β_i based on the cluster covariance matrix [**?**, **?**]. However, the new approach proposed in this paper uses a more compact representation. This representation is given by $\beta = \{(\rho_i, \alpha_i) \mid i = 1, \ldots, C\}$, where $\beta_i = (\rho_i, \alpha_i)$ are the polar parameters of the *i-th* straight line. From the polar representation of a line $\rho = x\cos(\alpha) + y\sin(\alpha)$ [**?**], a suitable choice for the distance function d is

$$d^2(x_j, y_j, \beta_i) = f^2(x_j, y_j, \beta_i) + g^2(x_j, y_j, \beta_i), \tag{4}$$

where $f^2(x_j, y_j, \beta_i) = (\rho_i - x_j\cos(\alpha_i) - y_j\sin(\alpha_i))^2$, and $g^2(x_j, y_j, \beta_i)$ is a penalty function for points that are very far from the cluster center.

Partitioning the data set $\mathcal{Z}$ into C prototypes β is accomplished by minimizing the objective function J

(eq. (1)). In order to consider the constraints (3), the Lagrange multipliers method is used and the problem becomes to minimize

$$V(\beta,\mathcal{U},\mathcal{Z}) = J(\beta,\mathcal{U},\mathcal{Z}) + \sum_{j=1}^{N} \lambda_j \left(1 - \sum_{i=1}^{C} u_{ij}\right), \quad (5)$$

where the λ_j's are the Lagrange's multipliers. V is minimized by using the alternating optimization method. Therefore, by considering β as constant, u_{ij} is determined as the one that satisfies $\partial V(\beta,\mathcal{U},\mathcal{Z})/\partial u_{ij} = 0$ and $\partial V(\beta,\mathcal{U},\mathcal{Z})/\partial \lambda_j = 0$. Thus, u_{ij} is given by

$$u_{ij} = \frac{1}{\sum_{k=1}^{C} \left[\frac{d^2(x_j,y_j,\beta_i)}{d^2(x_j,y_j,\beta_k)}\right]^{\frac{1}{m-1}}}. \quad (6)$$

In order to determine the parameters of the prototype β_i, the membership measures u_{ij} are considered as constants and the following equations set is solved: $\partial J(\beta,\mathcal{U},\mathcal{Z})/\partial \beta_i = \mathbf{0}$. Therefore, using eqs. (1) and (4), it can be shown that ρ_i and α_i are given by

$$\rho_i = \tilde{x}_i \cos(\alpha_i) + \tilde{y}_i \sin(\alpha_i), \quad (7)$$

$$\alpha_i = \frac{1}{2} \arctan\left(\frac{-2\tilde{S}_{xy_i}}{\tilde{S}_{yy_i} - \tilde{S}_{xx_i}}\right), \quad (8)$$

where

$$\tilde{x}_i = \frac{\sum_{j=1}^{N} u_{ij}^m x_j}{\sum_{j=1}^{N} u_{ij}^m}, \quad \tilde{y}_i = \frac{\sum_{j=1}^{N} u_{ij}^m y_j}{\sum_{j=1}^{N} u_{ij}^m}, \quad (9)$$

$$\tilde{S}_{xx_i} = \sum_{j=1}^{N} u_{ij}^m (x_j - \tilde{x}_i)^2, \quad \tilde{S}_{yy_i} = \sum_{j=1}^{N} u_{ij}^m (y_j - \tilde{y}_i)^2,$$

$$\tilde{S}_{xy_i} = \sum_{j=1}^{N} u_{ij}^m (x_j - \tilde{x}_i)(y_j - \tilde{y}_i).$$

In the above development, the penalty function g was kept as constant and given by

$$g^2(x_j, y_j, \beta_i) = (x_j - \tilde{x}_i)^2 + (y_j - \tilde{y}_i)^2, \quad (10)$$

where $\tilde{x}_i$ and $\tilde{y}_i$ are the weighted center of gravity coordinates of the *i-th* prototype from the last iteration (eq. (9)).

Figure 1: Simulation environment.

The prototype-based fuzzy line extraction algorithm is summarized as follows:

The main drawbacks of this algorithm are its sensitivity to initialization (e.g. the initial prototypes), to local minima of J, to noise and outliers and the difficulty to determine *a priori* the number of lines C. There exist other approaches to deal with these drawbacks as shown in [**?**].