Cooperating Robots
Lecture Notes
Topological Maps

Peter Johansen

26. februar 2001

The central data structure of the proposed design of a movable robot has been a table of features with information on their location and motion. In this note we shall consider an alternative design. We can characterize the difference between the previous classical design and the alternative one proposed here by the fact that the classical design has segmented the environment into separate features. The new design invokes the temporal succession of events and keeps a different data structure as basis for decisions.

What is an object? Is a pile of sand an object? Is a branch of a tree an object? It is difficult to give a satisfactory definition based on a single time instance of the scene in question. For a more satisfactory definition we need to appeal to a temporal sequence of images. In our case we have restricted ourselves to 1D images, but the argumentation applies to 2D images as well, although the features are much richer. A pile of sand is an object when unchanged over time, but is not when diffused by the wind. But an exact definition is still not easy to find.

Image sequences

Let us consider the sequence of images taken by a moving robot. Two images taken very close to another in time resemble each other. We later consider the case of occlusion and appearance of features. For now let the features in view be present in all images. We must leave the concept of an object. If one insists, one may think of an object as a section of space-time which possesses certain invariants. Invariants may be color, shape, spatial area, or other geometric or photometric invariants. But deciding whether a cloud is an object is still not easy. Let us expand the concept of object, which is a spatial concept, to a spatio-temporal event. We consider sections of space-time rather that objects.

Parameterization of a manifold

We consider space-time in relation to the observer. This means that we consider sections of the time series of images which hit the robot's CCD chip. Thus we do not make any attempt to detect distance to features or their coordinates as in the classical design. We try to engineer the system directly from the data. After all, an animal's retina receives a time series of images as well. We approach the inverse problem of constructing a representation of the surrounding world from the image data by constructing a *topological map*. By way of example consider a movie of a textured planar surface taken by an observer moving in 3D. Two images taken at consecutive time instances will only differ little. Let us construct a map of R^3 that at each point has an image as its value: the image taken by our hypothetical flying observer. The map will be continuous and even differentiable. Images taken at close postions resemble each other. If this 3D map is distorted by a diffentiable isomorphism of R^3 (a rubber transformation of R^3) the grid of 3D cartesian coordinates will be transformed into a 3D curvilinear grid. We notice that we can still use the position in the grid as a unique identification of the fly's position. Mathematically we express the scenery as follows: The different views (images) of the textured surface constitutes a 3D manifold. Really this is simply the definition of a a 3D manifold: a set (*in casu* of images) that can locally be parameterized by a 3D grid. In the example we can even use a global 3D grid. This is not possible if we want to parmeterize for instance the surface of a sphere. Here we need to compose the parameterization of two parameter grids. Usually we also by a manifold mean a set which is bounded and not as in our example an infinite set.

Algorithms

From an algorithmic point of view we can put a question as follows: Given a set of data (the images). Can we construct a mapping from a 3D grid with values in the set of images in such a way that close images have neighbor coordinates. If so, we can use the parameter values of an image as unique identification (coordinates) of the image. Or in other words, we have constructed an *index* into the image set. Algorithmically one usually assumes the grid's topology known. It can be a linear grid which consists of n points, a 2D $n \times n$ grid, 3D $n \times n \times n$ grid, or a 2D toroidal $n \times n$ grid to mention some examples.

The algorithm sketched above can indeed be implemented. The idea is to state the problem as an optimization problem. Perform an initial random 3D parameterization of the set of images, and transform it gradually into a better one. The criterion to be optimized is a measure of the differential of the map. For instance one could for each image compute the two-norm of the difference between the image and a neighbor image in the grid. If the sum of all differences is minimum, one can say that the parameter grid has been injected into the data sat in such a way that neighbor images resemble each other as much as possible.

Mentally one can think of a 2D parameterization of a data set as follows.

Image the data set ordered in a 2D grid in which neighbor elements occupy neighbor grid points. Take a handkerchief, crumble it, and throw it into the data set. The algorithm then decrumbles the handkerchief until it has been straightened out.

Several algorithms exist to do the job. One is *simulated annealing*, another is Kohonen's neural net algorithm [3]. I think much of Bruske and Sommer's algorithm [1]. See also Hafner [2] and Wichert [4]. The task is computationally complex.

Mobile robots

The above algorithm can be applied to the time sequence of images taken by a moving robot. The coordinates of the manifold will be two for determining the position of the robot and one to determine its orientation. In our arena, the parameter space will be the direct product of a 2D $m \times m$ grid and a circle with m points (a 1D torus).

We know a priori that such a parameterization exists. We can to each possible image assign its "real" coordinates and its orientation (assuming that no object in the environment moves).

Occlusion and appearance of features

Consider an image sequence taken by a robot, in which initially two features are present and later only one feature is present. The foreground feature will continue its motion at the time of occlusion. Its speed wil not be influenced by the background feature. Hence, it is possible to know which feature is closest at the time of occlusion. It can be verified by use of the infrared distance sensors. With reference to the previous discussion we will say that any two consecutive images in a sequence that contain an occlusion or an appearance are close to each other.

Environment includes a roaming robot

In principle it is easy to handle the situation in which the environment is no longer static, but for instance includes another robot. If we consider that robot a single feature and disregards its orientation, then the state of the environment requires two more parameters, the location of the other robot. Thus, the parameterization which the robot must compute has five parameters, a 1D torus (its orientation), a 2D grid for the static environment, and a 2D grid for the position of the other robot.

Difficulties

Suppose that we have computed the parameterization. Then the nearest neighbor among previously taken images to a new image taken by the robot can be located, and its parameterization read off. The parameters will tell the robot in which direction it looks, where it is located, and where the other robot is located.

There is of course an aber dabei: Much data is needed in order to cover this five dimensional parameter space. We should sample space so densely that two samples can decide if they are neighbors. In the case of insufficient data *aliasing* may occur. This means that there will be ambiguities, the new image has several close neighbors with deviating parameter values. More data may not even be enough to disambiguate the situation. For instance in the case of a very narrow camera field. Two images taken at different positions may be identical.

A remedy is to consider two consecutive images. Taken together, they reduce the ambiguity. By invoking still more predecessors, a unique identification may be found.

Micro events

We must consider the difficulty of determining the nearest neighbor to an image from a data base of images. This difficulty is not easily resolved. In the 2D case, an image contains so much information that an exact copy in the data base is out of the question. I have recommended that you reduce the difficulty by reducing the images to 1D. Let me first discuss the 2D case. A reasonable strategy would be to consider localized spatio-temporal events rather than sequences of entire images. This is the question of segmentation discussed above. Were it possible to parameterize localized spatio-temporal events, a sequence of entire images could be described by the events taking place, rather that by the parameters (global coordinates) of a parameterization of the entire image sequence. Then the robot would not know that it were at coordinates (x, y) but "only" that it had just turned a sharp corner and seen landmark z.

The animal visual system classifies localized spatio-temporal events. Area V1 in the visual cortex contains neurons that fire maximally when an edge moves across their receptive field with a specific speed and in a specific direction. The complete catalog of spatio-temporal events detected by the cortex is unknown. Much contemporary research is directed at constructing such a catalog for computational vision.

Let me again return to the 1D case. In order to cut the gordian knot, let me define a spatio-temporal event as an $n \times t$ segment of space-time. We recognize the restricted camera size from the discussion above. When we now slice the camera image into pieces, we must introduce yet another parameter: the position of the event in the image. But this will mean that we, given a static environment, can parameterize all the thus defined spatio-temporal segments by 4 parameters:

1. position (of the first spatial event of the segment) in the camera.

2. orientation of robot
3. two coordinates of the robot

When another robot is on the scene, we need an additional two parameters.

Let us consider the size of the segment. First, of course, there is an enormous number of different segments. More precisely, if all images are binary, there are $2^{n \times t}$ possible segments. Intuition tells us that the spatial extent of an event should not be less than 5-10 pixels. The temporal extent should not be less than maybe 5. This results in a minimum of $2^{25} \simeq 33\,000\,000$, much too high a number to expect an event to take place two times. This would be a correct argument **if** all events were equally frequent. Or in other words, if the robot only viewed random spatio-temporal events. This, of course, is not the case.

Event similarity

We need a criterion by which we can find the event in the visual memory of micro events that is most similar to a new one. This step is a tricky one. The question: How much are two images similar? cannot be answered in full generality. We have to simplify the problem. We have come part of the way be our construction of micro events as $n \times t$ segments of space-time. Bot as explained above, even if the parameters are chosen rather crudely as indicated, we may still have too many for hits to be frequent. It is difficult to say in advance. One could maybe expect that motion in our restricted universe is rather stereotype:

- a linear feature that moves in isolation. With $n = 10$ and 5 different speeds we get $\simeq 5 \times 10 = 50$ microevents. This would provide translation invariance. We will pick up motion at each spatial position in the event.
- two features that move each with its own speed. We get $\simeq 50^2 = 2500$ microevents.

A few thousand events is probably OK.

We may have to compute an equivalence between two images if identity does not suffice. Template matching is an obvious way: Compute the exclusive-or of the two micro event bit patterns. As the most similar event choose the event with most identical bits.

Event parameterization

Consider a static environment. Let us quantize the parameters

1. position (of the first spatial event of the segment) in the camera. 8 quanta.
2. orientation of robot. 5 quanta.
3. two coordinates of the robot. 10×10 quanta

This yields a total of $8 \times 5 \times 100 = 4\,000$ nodes in the parameter net. I have assumed that the events are non-overlapping in their spatial domain.

We should expect that certain micro events may be associated with several parameter values. An example is the all-zero event.

By the *ground truth* I understand the correct parameter values for each image in the time series and each position in the image. This way each combination of parameter values points to a micro event. On the other hand, a micro event may be pointed at by several sets of parameters. This may be satisfactory if a suitable voting rule can be designed by which the micro events in a new image can decide where they are. The micro events are sufficiently well designed if only one position is associated with them all. Realistically, each micro event votes for its possible positions and a majority decision is taken. Let us say that the time series is *sufficient* if the position of the robot can be recovered from the micro events.

Once he micro events have decided the location of the robot, they can try to decide their position in the camera image. It may be satisfactory thatthese positions can only be partially determined. Thus a micro event may have taken place at two different image positions.

Project proposal

The proposal suggested below is quite ambitious. Certainly suitable for a joint effort. Please consider forming groups to realize this project.

Using the infrared remote control, direct the robot around in the scene which has been suitably marked with landmarks. Collect a time series of 1D images. Notice the position each time a photo is taken. This info serves as ground truth. Choose n and t sufficiently large to disambiguate the ground truth map. Compute the parameterization of the spatio-temporal events. The computations should be made on-line in the sense that after t images the data structures should be updated. This will mean increasing precision in the predictions. Success criterion: Compute the isomorphism (rubber map) between the ground truth parameterization and the computed parameterization. Direct the robot by the infrared remote control and let it transmit its floor coordinates [1] to the PC for display on an image of the arena. Expected outcome: Initially the circles marking its position will be randomly scattered, but the number of hits will become still more frequent.

To realize this project we need to have access to tracking information from the ceiling-mounted camera. We need to have the radio transmission working. We need to transmit the image sequence to the PC. We need to compute a parameterization of the micro events. We need to compute the nearest micro event. We need to decide a voting procedure for micro events.

It is very easy to think of extensions. Let 2D images replace 1D binary signals, or/and introduce another robot on the scene.

[1] computed the hard way, of course

Litteratur

[1] J. Bruske and G. Sommer. An algorithm for intrinsic dimensionality estimation. In G. Sommer, K. Daniilidis, and J. Pauli, editors, *Computer Analysis of Images and Patterns*, volume 7th Int. Conf., CAIP'97, pages 9–16. Springer-Verlag, 1997.

[2] Verena Hafner. Learning places in newly explored environments. In Meyer, Berthoz, Floreano, Roitblat, and Wilson, editors, *SAB2000 Proceedings Supplement Book*. International Society for Adaptive Behavior, 2000. postscript at http://www.ifi.unizh.ch/groups/ailab/publications/2000.html.

[3] T. Kohonen. Clustering, taxonomy, and topological maps of patterns. In *Proceedings of the 6th International Conference on Pattern Recognition*, October 1982.

[4] Georg von Wichert. Selforganizing visual perception for mobile robot navigation. In *1st EUROMICRO Workshop on Advanced Mobile Robots (EUROBOT'96)*, 1996.