

Cluster Tracking with Time-of-Flight Cameras

Dan Witzner Hansen

Mads Syska Hansen

Martin Kirschmeyer

Rasmus Larsen

Davide Silvestre

Informatics and Mathematical Modelling

Technical University of Denmark

Abstract

We describe a method for tracking people using a Time-of-Flight camera and apply the method for persistent authentication in a smart-environment. A background model is built by fusing information from intensity and depth images. While a geometric constraint is employed to improve pixel cluster coherence and reducing the influence of noise, the EM algorithm (Expectation Maximization) is used for tracking moving clusters of pixels significantly different from the background model. Each cluster is defined through a statistical model of points on the ground plane. We show the benefits of the Time-of-Flight principles for people tracking but also their current limitations.

1. Introduction

Tracking multiple people is key in surveillance applications. A significant number of methods for tracking people in camera images have been proposed [6, 10], addressing challenging issues such as occlusion, close interactions, shadows and pose estimation. Prior models of the objects being tracked and geometry of the scene can resolve some of the problems found when the scene is viewed from a single camera. Stereo may simplify the tracking algorithms and occlusion issues due to the depth information, but it also introduces new issues: stereo calibration, additional space required for several cameras and unobstructed views in both cameras. In addition to these stereo also requires non-trivial and time-consuming processing of the images to determine point correspondences.

Time-of-Flight sensors are relatively new and novel developments in imaging devices, providing real-time gray scale and dense depth information from a single sensor. Compared to standard stereo vision, Time-of-Flight sensors offer several advantages as only a single sensor is required. Calibration between the cameras or additional image processing (correspondence matching) for obtaining depth measurement is obviated. Compared to structured light methods the Time-of-Flight cameras emit much less

light, thus neither the image nor a person facing the camera are disturbed. The major limitations of current Time-of-Flight sensors are the reduced resolution (176×144 for the SwissRanger), only gray scale images are generated, and that depth measurements may be influenced by the reflective properties and color of the object [4]. The depth information makes some applications more robust despite the low resolution and by combining Time-of-Flight cameras with high resolution cameras the benefit of both methodologies may be exploited [5]. Time-of-Flight cameras have been employed for multiple people tracking. The method proposed by Bevilacqua *et al.* [1] use a top-down view of the scene. The measured heights from the ground plane is used directly as an indicator of the presence of a person. Using a top-view of the ground floor simplifies tracking, but it also limits applicability as only a smaller portion of the scene is visible. Besides, useful information about the person is hidden due to self-occlusion.

We propose a method for tracking people moving on a planar surface with images obtained from a (mostly stationary) Time-of-Flight camera. Potential objects in the foreground are segmented through the difference of the current image to a statistically learned background model image (section 2). Objects on the ground floor are mapped through a homographic mapping to a top-view image as to obtain Euclidean information. As described in section 3, not only do homographies map points from projective planes, but they also contain information of the normals to the plane. We use depth information and the plane normal for filtering noise in the depth image as well as to enhance connectivity of pixel clusters on the ground plane. The final steps of the method track the individual cluster pixels under a Gaussian model in the image. We employ a statistical model of image clusters and use the EM algorithm for cluster parameter estimation described by Pece [8]. The tracking model is described in section 4. A major benefit of the tracking model is the use of thresholds becomes needless. The tracker is employed within a smart environment where smart cards are used for identification of the person. Several identification areas are located within the smart environment (de-

fined in a top-view map). A person is only granted access to secured areas when being identified. Often repetitive authentications are needed when performing frequent access control. The idea of persistent authentication is to automate repetitive tasks of authentication. People tracking is one way towards persistent authentication. An evaluation of the tracker targeting persistent authentication is given in section 5 and we conclude the paper in section 6.

2. Background Modeling

Background modeling is core in most tracking methods where the camera is stationary [2, 9]. The principle behind these models is that the pixels belonging to the background are stable and do not vary significantly. Foreground objects are detected in the image obtained from subtracting the background model image with the current image. Simple frame differencing is clearly not sufficient but robust statistical methods are required to cope with noise from the camera, background changes such as moving trees and fluorescent light. The background model needs to be updated regularly. Methods relying on standard cameras face problems when the foreground object has a similar color as the background. Fusing depth information into the background model should therefore accommodate changes in both intensity and depth and handle instabilities caused by the light changes, presence of shadows and people appear similar to the background.

Background Models The fundamental idea of background models is to represent the distribution of intensity values over time either through parametric or non-parametric density models. In this paper we explore both approaches through a mixture of Gaussian [9] and kernel density estimation models [3] and compare their efficiency. We build a background model on the joint distribution of intensity and depth information.

Mixture of Gaussians (MoG) are parametric models for density estimation and have been explored for background modeling by Stauffer and Grimson [9]. The probability of a point x_t to be background at time t is under this model defined as:

$$P(x_t) = \sum_{i=1}^K \omega_i N(x, \mu_i, \Sigma_i) \quad (1)$$

where K is the number of Gaussian (N) probability density functions, ω_i , μ_i and Σ_i are the weight, mean and the covariance matrix, respectively, of the i 'th mixture. The model is updated at each frame using the EM algorithm.

Kernel density estimator are non-parametric density models where no direct assumptions on the functional structure are made [3]. For each location the classification depends on the values that the pixel has had in the last N im-

ages. The probability that a pixel has value x_t at time t with kernel Ψ :

$$P(x_t) = \frac{1}{N} \sum_{i=1}^N \Psi(x_t - x_i) \quad (2)$$

We use the Gaussian kernel Ψ giving the kernel density estimate:

$$P(x_t) = \frac{1}{N} \sum_{i=1}^N \frac{1}{(2\pi)^{\frac{d}{2}} |\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(x_t - x_i)^T \Sigma^{-1} (x_t - x_i)} \quad (3)$$

Comparison Each method is evaluated by varying their parameters (*e.g.* the number of clusters, learning rate and number of frames in) as to minimize false positives and maximize correctly classified pixels. The parameters of each model (KDE and MoG) have been individually optimized. A set of 7 different test scenarios with varying number of people, w/o occlusion, light condition and movements speeds) totalling 996 frames of which 70 frames have been manually segmented. The comparison for the best performing parameters yields quite similar number of correctly classified pixels (53.7% for KDE and 57.4% for MoG). Also the average misclassification rates are similar but with an average of 60 pixels better performance for the MoG model. The MoG model requires less computation and memory since only the model parameters are updated base on a single frame rather than on the entire history. Based on the marginal better performance we opt for the MoG model in the subsequent steps.

3. Geometric Clustering

Tracking people from a top-view is generally easier than from oblique angles since occlusion occurs infrequently, clusters of points belonging to a person is connected and circular. It is also straight forward to obtain Euclidean information. The limitation of top view images is their limited field of view. Homographies are plane to plane projective transformations and common to use for mapping the location of a person from a camera view to a top-view image (*e.g.* an overview map). We use a manually selected homography to ensure that points on the ground plane are mapped correctly to the overview image. A focal length normalized homography contains information about the ground plane normal [7]. The normal can be used to project off the ground plane to the ground plane and is useful for people tracking since walking and standing people are roughly perpendicular to the ground plane [6]. Projecting points along the normal using standard cameras is not reliable since a particular connected set of points in the image may come from several depths. Since Time-of-Flight cameras provide this information it is possible to project points onto the ground plane, distributing the points on the ground plane according to the depth as depicted in Figure 1. The map of

projected points is called *flat-map*. Similar steps are found in Harville [6]. Consequently, points that are coherent in 3D space are connected in the *flat-map*, δ . If noise is present then this will be distributed along the depth axis and present less clutter on the ground plane. Similarly relative heights are implicitly given through the normal.

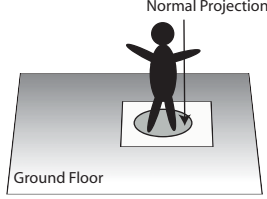


Figure 1. Using the normals obtained from the homography ensure better coherence of people clusters.

4. Cluster Tracking

The tracking model is based using EM for maximum likelihood estimation of the parameters of a mixture of Gaussian clusters in the flat-map. The cluster model is inspired by Pece [8]. The mixture consists of n foreground mixtures and one background cluster. The probability models for foreground ($j > 0$) and background clusters ($j = 0$) are of course different. Each cluster $\lambda_j = (c_j, \Sigma_j, \mu_j, \omega_j)$ is defined through the centroid and covariance of the cluster, the mean intensity and the percentage of pixels generated from cluster j , respectively.

Initially the method detects new clusters according to the information coming from the previous frame and the cluster parameters are updated using the EM algorithm. Each cluster is given by a mean and covariance structure and are therefore ellipsoidal. After having found new ellipses and their parameters updated, the clusters are analyzed and according to their parameters to whether they could be deleted or merged.

4.1. The Cluster Model

The probability f_j that cluster j generates a pixel value at the location u is divided into two components:

$$f_j(u) = g(u|\lambda_j) \cdot h(\delta(u)|\lambda_j) \quad (4)$$

where g depends on the coordinates of the image and h on the intensities in the flat-map (δ). The definition of the functions are described below.

Background cluster The background is defined in all pixel locations (there is always a background even though it is occluded by persons) and is treated as being uniformly

distributed:

$$g(u|\lambda_0) = \frac{1}{m} \quad (5)$$

where m is the number of image locations.

Notice that the background cluster is defined after the foreground and background images have been subtracted and therefore should not vary significantly. Small values of the image indicate higher likelihood of belonging to the background and can thus be modeled through a Gaussian:

$$h(pr(u)|\lambda_0) = \frac{1}{2\mu_0} \exp\left(-\frac{|\delta(u)|}{\mu_0}\right) \quad (6)$$

where μ_0 is the absolute mean of the values within the background cluster.

Target clusters For the target clusters the function h is considered uniformly distributed:

$$h(u|\lambda_j) = \frac{1}{q} \quad (7)$$

where q is the number of observable probability values.

The distribution g depends on the distance between the pixel and the cluster centroid and is considered normal:

$$g(u|\lambda_j) = \frac{1}{2\pi\sqrt{|\Sigma_j|}} \exp\left(-\frac{1}{2}\Delta u_j^T \cdot \Sigma_j^{-1} \cdot \Delta u_j\right) \quad (8)$$

where $\Delta u_j = u - c_j$ is the vector distance between the pixel and the centroid of the cluster j and Σ_j is the covariance matrix of the cluster.

The posterior probability that a point belongs to the cluster j is given by:

$$p_j(u) = \frac{w_j f_j(u)}{f(u)} \quad (9)$$

where $f(u)$ the prior probability of the point:

$$f(u) = \sum_{j=0}^n w_j f_j(u) \quad (10)$$

Cluster detection Clusters are detected through local maxima in a down-sampled flat-map image (by factor L) weighted by the probability of belonging to the background. Only local maxima in the background not already detected are therefore found:

$$pr_0(u) = pr(u)p_0^{(t,0)}(u) \quad (11)$$

where $p_0^{(t,0)}(u)$ is an estimate of $p_0(u)$, obtained in the last iteration of the previous frame. Local maxima satisfying the following relation are kept:

$$p\hat{r}_0(u) > \mu_0(1 + \log \frac{q}{2\mu_0}) \quad (12)$$

where μ_0 is the background average and q the number of possible values image values. Notice that the values in the threshold are all given directly from the model. This values comes from the cost of merging two clusters. Based on this we should emphasize that detection of new clusters do not require thresholds, because the minimum ratio μ_j/μ_0 to generate a cluster is expressed as $1 + \log \frac{q}{2\mu_0}$.

Define the pixel density within a cluster:

$$\zeta_j = \frac{mw_j}{2\pi\sqrt{|\Sigma_j|}} \quad (13)$$

Since the background is assumed stationary, values inside the background cluster do not change significantly:

$$\zeta_0 = w_0 \quad (14)$$

Define the cost of merging clusters λ_0 and λ_j :

$$M(j, 0) = mw_j \left[\log \frac{\zeta_0}{\zeta_j} + \log \frac{q}{2\mu_0} - \frac{\mu_j}{\mu_0} + 1 \right] \quad (15)$$

The ratio $\frac{\zeta_0}{\zeta_j}$ tend to remove clusters with low density and $\frac{\mu_j}{\mu_0}$ maintains clusters with regions of high probability and it follows that when $M(j, 0) > \theta_M$:

$$\frac{\mu_j}{\mu_0} \geq 1 + \log \frac{q}{2\mu_0} + \log \frac{\zeta_0}{\zeta_j} + \frac{\theta_M}{mw_j} \quad (16)$$

The result in equation 12 follows by neglecting the last two terms.

Merging clusters Due to the background model and noise it may happen that objects are split into separate clusters. Two clusters are merged provided the resulting cluster remains ellipsoidal and the two cluster are close according to both Mahalanobis distances $D_{\Sigma_j}(i, j)$ and $D_{\Sigma_i}(i, j)$.

Eliminating clusters The criteria to eliminate clusters is based on comparing the average of the cluster with the average of the background as well as the dimensions of the ellipse. This test is performed at each iteration of the EM algorithm because the method has a complexity linear with the number of clusters and for that reason it is convenient to remove clusters as soon as possible.

A cluster is eliminated if at least one of following conditions is satisfied.

1. the average absolute value of the probability image for the cluster j is smaller of the background average multiplied by a constant κ_μ

$$\mu_j < \kappa_\mu \mu_0 \quad (17)$$

2. The weight (prior probability) of the cluster, w_j , is less than L^2/m , where L is the cell size used to down-sample the image during the detection of new clusters and m is the dimension of the image.

4.2. Parameters estimation

The Expectation-Maximization (EM) algorithm is used to maximizing log-likelihood of the cluster parameters and is commonly used for people tracking:

$$\hat{L}(\lambda_j|D) = \sum_u \sum_j p_j(u) \log(w_j f_j(u)) \quad (18)$$

EM alternates between the expectation step, which computes an expectation of the likelihood by including the latent variables as if they were observed, and a *maximization* step, which computes the maximum likelihood estimates of the parameters by maximizing the expected likelihood found on the expectation step. The parameters found in the maximization step are then used in the subsequent expectation step. The process is repeated until convergence. This leads to the following update steps for the k th iteration [8]:

$$\mu_j^{k+1} = \frac{\sum_u |pr(u)| \cdot p_j^{(k)}(u)}{\sum_u p_j^{(k)}(u)} \quad (19)$$

For the covariance matrices the ML estimate ($\hat{\Sigma}$) is weighted with a factor $1/\tau_\sigma$:

$$\Sigma^{(f,i)} = \Sigma^{(f-1,\infty)} + \frac{1}{\tau_\sigma} (\hat{\Sigma}^{(f,i)} - \Sigma^{(f-1,\infty)}) \quad (20)$$

where f is the index of the frame, i the number of the iteration and ∞ the index of the last iteration in the previous frame. Besides the ML estimate has been calculated as:

$$\hat{\Sigma}^{(f,i)} = \frac{\sum_u (u - c)(u - c)^T \cdot p^i(u)}{\sum_u p^i(u)} \quad (21)$$

denoting $p^i(u)$ is the posterior probability at the i 'th iteration and c the centroid of the cluster.

When a new cluster is detected, Σ is initialized with an initial guess corresponding to a circular region. The cluster is subsequently allowed to grow according to the values of $p^i(u)$. In fact if a point u is close to the cluster λ_i with a high probability belonging to the foreground means that $f_i(u)$ is greater than $f_0(u)$ and thereby the posterior probability $p_i(u)$ is large, allowing the cluster i to grow and cover u .

Since the estimate of the background cluster parameters are hard to compute and they are not significantly affected by the changes of the other clusters, it is sufficient to update them only once after the convergence of the EM algorithm.

5. Evaluation

In the following section the tracking method is evaluated, but with a persistent authentication in mind. A set of 23 different test scenarios containing a total of 6770 image tuples (gray scale and depth images). Several issues arise in

persistent authentication when using cameras, namely *persistence* (maintain the authentication of the users) and *robustness* (prevent clearance to be usurped by another user). For example identity swaps may occur when several persons are in the same smart environment (*e.g.* occlusions). Similarly, tracking should be maintained when either the appearance of the person (*e.g.* changing clothes) or the surrounding changes (*e.g.* lights on/off) or someone is trying to actively compromise security though their appearance or visual disturbances. Persistent authentication may relax the assumptions of the tracking algorithm by noticing that being able to track an authenticated person reliably is more important than tracking all persons in the view of the camera. We assume that a person leaving the field of view and re-entering are separate persons.

An important property of Time-of-Flight cameras is their ability to operate in dark environments. Figure 2 shows an example of a dark room where the Time-of-Flight camera is capable of tracking a single person. In this particular sequence the only falsely detected cluster is caused by haloing effects when the person enters the scene close to the camera.



Figure 2. Tracking a person in a dark room. left) gray image, middle) depth image and right) plane view with cluster ellipse of the person. The blue overlay region indicates the location of a door.

Occlusions may lead to mistaken identities when two or more persons are present in the same room. Maintaining track during and after occlusions are key for reliable authentication. Figure 3 and figure 4 show samples from two sequences where the persons tracked despite close contact and occlusion. In these sequences the detection and tracking accuracy is 98%. The main causes of the errors (14/591 frames) are due to the persons moving in front of the door. When moving in front of the door, the camera adapts the light emission to the particular scene. When the persons move away from the door reflections will appear and cause a cluster to be detected (see figure 4).

A main advantages of the Time-of-Flight technology is that they can operate in both dark and light conditions, but the use of IR light may also be their demise. Figure 5 shows that significant errors are encountered when Time-of-Flight cameras are pointed towards each other. The disturbances are magnified when the person is approaching the stationary camera.

In conclusion, even in the presence of occlusions the use of Time-of-Flight cameras produce fairly robust and reliable results. The method is capable of tracking people walking

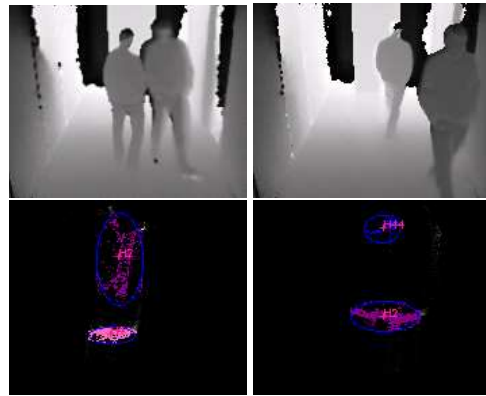


Figure 3. Two persons crossing. Top) Depth image sequence (frame numbers 114,118 and 125). Bottom) Corresponding plane view images with tracked clusters.

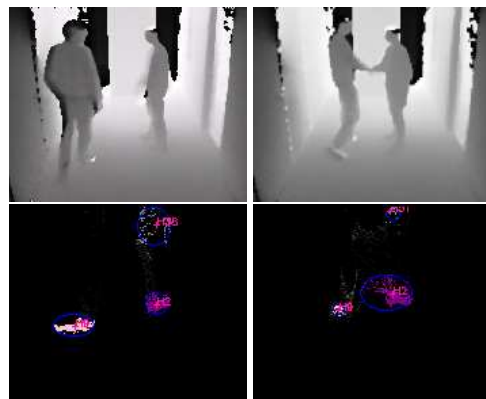


Figure 4. Handshake sequence. An extra cluster is detected due to reflections on the door in the background. Top) Depth image sequence. Bottom) Corresponding plane view images with tracked clusters.

with an accuracy of 98%, but very rapid movement like running and jumping and other Time-of-Flight cameras seem to confuse the camera measurements and cause erroneous clusters to be detected. The majority of the false positive encountered are caused by incorrect camera illumination causing haloing effects and reflections in the background.

6. Discussion

We have described a model for people tracking in Time-of-Flight images using statistical models of clusters and background. Tracking is performed on the ground plane by projecting points according to the plane normal. In this way depth is used both to cluster coherent points as well as filtering noise. The influence of noise is also handled by the statistical model of clusters over time. Occlusions can therefore be handled more effectively since tracking is performed directly on a 3D plane. In general tracking per-

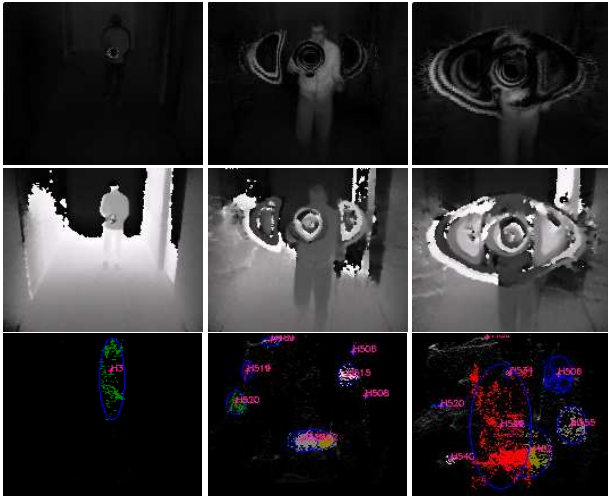


Figure 5. Disturbances from another Time-of-Flight camera(frames 114,118 and 125). Top) Gray scale image sequence, Middle) Depth image sequence . Bottom) Corresponding plane view images with tracked clusters.

formance is good and the use of a robust background model using both gray scale and depth images is important for stability of the method. We obtain a detection accuracy in the frames about 98% under normal walking conditions. Improvements by using Time-of-Flight sensors are revealed when tracking is exposed to dark environments and when using the depth information to limit redundancy in pixel intensities and in handling the relative high amounts noise present in the Time-of-Flight images. Even in the presence of occlusions, the method is capable of maintaining track. The main reason is that tracking effectively use depth information over time for disambiguation. The main causes of errors are due to reflective properties of the background, fast moving objects (jumping and running) as well as other Time-of-Flight cameras in the scene.

Time-of-Flight sensors should during day light conditions provide similar results as a stereo rig with low resolution images, but they avoid stereo calibration require less space. A limitation of the Time-of-Flight sensor is that the the depth measurements are affected by the surface materials in the scene. Reflections can cause depth measurement errors and even though a background model is present, false detections occur.

Tracking performance using Time-of-Flight cameras is limited by the low resolution, but even though the technology seems promising for at least applications like persistent authentication and surveillance. The current model does not distinguish between object types (such as humans, cars etc) and future work may reveal methods handling these cases and hopefully reduce errors further.

Acknowledgements

This work has partially been funded by the European Commission (contract no. IST-34107) within the Information Society Technologies (IST) priority of the 6th Framework Programme (ARTTS project). This publication reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

References

- [1] A. Bevilacqua, L. Di Stefano, and P. Azzari. People tracking using a time-of-flight depth sensor. *IEEE International Conference on Video and Signal Based Surveillance, 2006.*, pages 89–89, 2006. [1](#)
- [2] T. Darrell C. Wren, A. Azarbayejani and A.P. Pentland. Pfinder: Real-time tracking of human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:780–785, 1997. [2](#)
- [3] Ahmed M. Elgammal, David Harwood, and Larry S. Davis. Non-parametric model for background subtraction. In *ECCV '00: Proceedings of the 6th European Conference on Computer Vision-Part II*, pages 751–767, London, UK, 2000. Springer-Verlag. [2](#)
- [4] Dragos Falie and Vasile Buzuloiu. Noise characteristics of 3d time-of-flight cameras. In *IEEE Sym. on Signals Circuits & Systems (ISSCS)*, 2007. [1](#)
- [5] Sigurjon Guðmundsson, Henrik Aanaes, and Rasmus Larsen. Fusion of stereo vision and time-of-flight imaging for improved 3d estimation. *Int. J. of Intelligent Systems Techn. and App. Issue on Dynamic 3D Imaging*, In press, 2007. [1](#)
- [6] Michael Harville. Stereo person tracking with adaptive plan-view templates of height and occupancy statistics. *Journal of Image and Vision Computing*, pages 127–142. [1](#), [2](#), [3](#)
- [7] Yi Ma, Stefano Soatto, Jana Kosecka, and S. Shankar Sastri. *An Invitation to 3-D Vision: From Images to Geometric Models*. Springer-Verlag, 2003. [2](#)
- [8] Arthur E. C. Pece. Generative-model-based tracking by cluster analysis of image differences. *Robotics and Autonomous Systems*, 39(3–4):181–194, 2002. [1](#), [3](#), [4](#)
- [9] C. Stauffer and W.E.L. Grimson. Adaptive background mixture models for real-time tracking. *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, 2:246–252, 1999. [2](#)
- [10] Christopher Richard Wren, Ali Azarbayejani, Trevor Darrell, and Alex Pentland. Pfinder: Real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):780–785, 1997. [1](#)