Saliency Detection and Model-based Tracking: a Two Part Vision System for Small Robot Navigation in Forested Environments

Richard Roberts*, Duy-Nguyen Ta, Julian Straub, Kyel Ok, and Frank Dellaert
Center for Robotics and Intelligent Machines, Georgia Institute of Technology

ABSTRACT

Towards the goal of fast, vision-based autonomous flight, localization, and map building to support local planning and control in unstructured outdoor environments, we present a method for incrementally building a map of salient tree trunks while simultaneously estimating the trajectory of a quadrotor flying through a forest. We make significant progress in a class of visual perception methods that produce *low-dimensional, geometric information* that is ideal for planning and navigation on aerial robots, while directing computational resources using *motion saliency*, which selects objects that are important to navigation and planning. By *low-dimensional geometric information*, we mean coarse geometric primitives, which for the purposes of motion planning and navigation are suitable proxies for real-world objects. Additionally, we develop a method for summarizing past image measurements that avoids expensive computations on a history of images while maintaining the key non-linearities that make full map and trajectory smoothing possible. We demonstrate results with data from a small, commercially-available quad-rotor flying in a challenging, forested environment.

1. INTRODUCTION

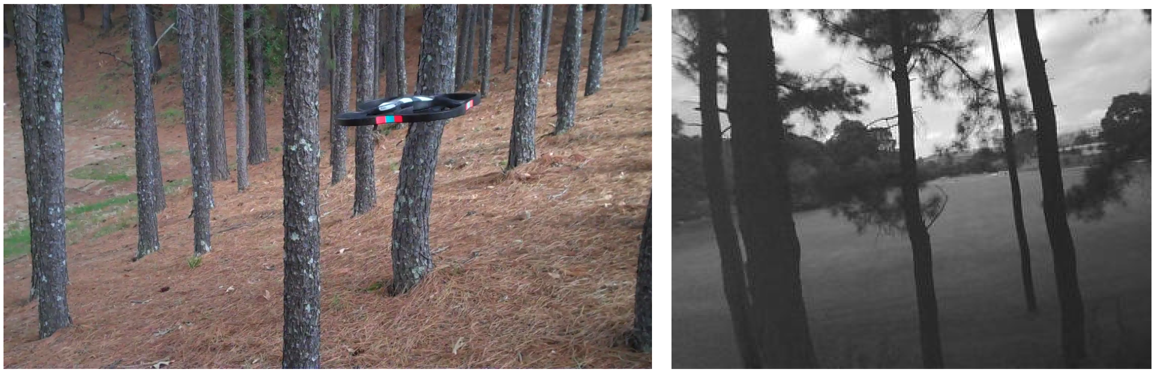
We are working towards the goal of fast, vision-based autonomous flight, localization, and map building to support local planning and control in unstructured outdoor environments. With its wide range of important applications in both military and civilian services, research in Unmanned Aerial Vehicles (UAVs) has been growing significantly in recent years. Despite many common characteristics, the problems of automatic navigation, obstacles avoidance, map building, and action planning in an aerial robot are much harder than in a typical ground robot. This is mainly because of the limited payload, limited power consumption, and the extra degrees-of-freedom motion of an aerial robot.

In the context of a vision-equipped quadrotor flying through a forest, we present a method for incrementally building a map of salient tree trunks while simultaneously estimating the quadrotor's trajectory. Towards this overall goal, we make significant progress in a class of visual perception methods that produce *low-dimensional, geometric information* that is ideal for planning and navigation on aerial robots, while directing computational resources using *motion saliency*, which selects objects that are important to navigation and planning. By *low-dimensional geometric information*, we mean coarse geometric primitives, which for the purposes of motion planning and navigation are suitable proxies for real-world objects.

Mapping tree trunks in a forest from a flying quadrotor, and more generally perception from small robots in many real-world outdoor scenarios, are difficult problems due to platform limitations and challenging properties of the environment. The limited payload capacity and limited power of small aerial robots makes it difficult to use laser scanners and cameras with large, high-quality lenses. Furthermore, while current state-of-the-art autonomous mapping and navigation methods build 3D point clouds from point image features, such features often do not appear, are unstable, or are not discriminative enough for matching on important objects in many real-world situations when viewed through miniature, lightweight cameras. As an example of this, compare the photograph from a high-quality camera in Fig. 1a to the view from the quadrotor's on-board miniature camera in Fig. 1b. Additionally, computational power as well as wireless network bandwidth make processing on-board or transmitting high-quality images very challenging or impossible.

Our method specifically addresses the limited computational resources of small aerial robots. It improves upon the state-of-the-art by directly estimating low-dimensional geometric information in the form of vertical cylinders modeling tree trunks as well as the robot's trajectory. This avoids the aforementioned problems with perceiving point clouds, making it possible to track featureless edges, as well as produces geometric information about occupied space that is immediately useful to a motion planner or controller. Furthermore, we use a bio-inspired measure of *motion saliency* to limit mapping to trees that are nearby to the robot, and thus important for motion planning. Additionally, we develop a method for summarizing past image measurements that avoids expensive computations on a history of images while maintaining the key non-linearities that make full map and trajectory smoothing possible.

*richard.jw.roberts@gmail.com



(a) Photograph of the environment taken by a high-quality camera.

(b) View of the environment from the quadrotor's on-board lighweight, miniature camera.

Figure 1: Point image features often do not appear or are not discriminative enough for matching in many real-world situations when viewed through the miniature cameras that are suitable for lightweight aerial robots. Our method targets such platforms, and instead directly estimates low-dimensional geometric information that is immediately useful to a motion planner. We present results in this forest setting, mapping tree trunks and estimating the quadrotor's trajectory, with a commercially-available and inexpensive UAV, the Parrot AR.Drone.

2. RELATED WORK

Many systems use laser scanners on the UAV to navigate in indoor[1–3] or outdoor[4] environments. Unfortunately, laser scanner is not suitable for long-term use on small aerial robots due to their limited power and payload capacity. Moreover, in many applications (e.g., military) active sensors, such as laser scanners, are to be avoided. For example, multiple laser scanners can interfere with each other and are not suitable for covert operation.

Our interest in lightweight UAVs precludes the use of heavy and power-hungry sensors such as laser scanners. Instead, we target *vision* sensors combined with an IMU, due to their lightweight, low-power and wide availability in popular UAVs like the AR.Drone. Typical work utilizing the light-weight on-board camera for navigation either relies on a ground robot for map-building,[5] or performs full SLAM using the downward camera capturing the ground.[6–9] Unfortunately, the map of the ground obtained from the downward camera is insufficient for obstacle avoidance task. Many other work utilizes the Parallel Tracking and Mapping Algorithm (PTAM)[10] to build point-cloud based maps of the environment.[7–9] This imposes an area restriction, since PTAM is limited to a small workspace.[10] The goal of Langelaan and Rock[11] to enable autonomous UAV flight in the forest using only light-weight sensors like IMU and camera is most closely related to ours. Unfortunately, their paper[11] only demonstrates results on ground robots with synthetic data or balloons as tree replacements.

Model-based object tracking for localization has also been shown to be a powerful tool in aerial robots. Kemp[12] uses a 3D model of the environment, projects that onto the image and optimizes the pose parameters through a likelihood function of the SSD distances from sampled points on the projected edges to nearby edges. One key to robustness during fast motions was to retain multiple hypotheses about which image edge corresponded to each model edge. Similarly, Teulière *et al*.[13] also uses model-based edge tracking for indoor flight.

Unfortunately, these edge-based methods only work well in a controlled indoor environment with a small number of easily detectable edges. In our forest situation, binary edges such as that of Canny,[14] are undesirable since they might contain many unwanted edges from leaves and branches in the forest while discarding lots of useful information from the original image measurements. Hence, instead of detecting edges in the image, our model-based approach uses a likelihood function to maximize image gradients along the tree boundaries to estimate the 3D geometry structure.[15–17]

Our work is inspired in part by studies on the visual and spatial reasoning systems of animals including humans. Psychologists know that optical flow sensation occurs early in the visual pipeline, and Gibson proposed that optical flow provides crucial information for animals to perceive and avoid looming obstacles, pursue prey, and perform a number of other higher-level tasks.[18] Optical flow reasoning over the entire visual field, similar to the peripheral flow we model, allows perception of ego-motion and basic control laws such as avoiding obstacles, following walls and tunnels, and

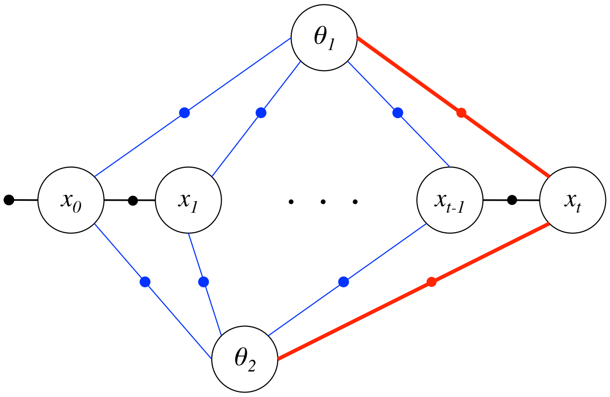


Figure 2: Factor graph for the tracking system, where x_t is the pose of the Ar.Drone at time t and θ_j comprises the parameters of the jth tree. The current original measurement factors (red thick edges) will be turned into the approximate summary factors (blue thin edges) after optimization. Consecutive poses are connected by odometry measurement factors.

landing. Besides optical flow, humans and other animals additionally keep track of the positions of distinct objects.[19] This information about individual objects is complementary to coarse, wide-field optical flow.

The peripheral-foveal system of Gould *et al.*[20] shares with us many similar ideas of combining peripheral and fovea tracking mechanisms, but their goal is object recognition and tracking, whereas ours is map building and outdoor navigation on UAVs. We also note the classic work of Itti *et al.*[21] which proposes a computational model for the visual attention system. Furthermore, another paper of Itti and Koch[22] also reviews different related models for the saliency map, attention and eye movements, and the inhibition of return for scene understanding and object recognition.

3. MAPPING THE FOREST

Our goal is to map *salient* trees, i.e. those that are nearby to the robot. We want to infer the planar positions and radii $\theta_j \in \mathbb{R}^2 \times \mathbb{R}^+$ of salient trees (indexed by j) as well as the trajectory of the robot described by the poses $x_t \in \mathbb{SE}(3)$ at every time step t. All positions and poses are expressed in a global coordinate frame. We obtain measurements z_t of the robot's odometry from an on-board pose filter, and image measurements I_t at each time step. Specifically, we wish to infer

$$p\left(x,\theta\,|\,I,z\right), \tag{1}$$

where x is the set of all poses, θ the set of all *salient* trees that have been detected, z the set of all odometry measurements, and I the set of all images.

We decouple the detection and tracking problems. During inference of the map and trajectory, we assume that the set of salient trees observed in each frame is known. At every new time step, we propagate new observations of all trees that were observed in the current time step. Additionally, we formulate a model-based tree detector, making use of *motion saliency*, to instantiate newly-observed trees.

4. INFERRING THE TREE PARAMETERS AND ROBOT TRAJECTORY

We formulate this estimation problem as a factor graph comprised of pose nodes x_t and tree nodes θ_j, which are the variables we want to estimate. Those nodes are connected via three different types of binary measurement factors, which represent constraints between the respective variables: odometry measurement factors connecting two poses, image measurement factors connecting a pose and a observed tree, and summary factors, which are the aforementioned approximations of the image factors. The formulation of this problem as a factor graph allows us to solve for the unknown variables using the GTSAM library.[23]

We assume that an image measurement I_t depends only on the robot pose x_t and tree parameters θ_t from the same time step, and that an odometry measurement z_t depends only on the previous and current[†] poses x_{t-1} and x_t. Using these conditional independence assumptions, Eq. 1 factors as

$$p\left(x,\theta\,|\,I,z\right)\propto\prod_{t=1}^{T}p\left(I_t\,|\,x_t,\theta\right)\prod_{t=2}^{T}p\left(x_t\,|\,x_{t-1},z_t\right)p(\theta)p(x_1), \tag{2}$$

where $p\left(I_t\,|\,x_t,\theta\right)$ is the image measurement model of how likely an image I_t is given a robot pose x_t and set of trees θ, $p\left(x_t\,|\,x_{t-1},z_t\right)$ is a motion model to predict the current pose x_t from the previous pose x_{t-1} given the current Gaussian-noise odometry measurement z_t, and $p(\theta)$ and $p(x_1)$ are priors on the tree parameters and the first pose respectively. In our experiments, $p(x_1)$ serves only to anchor the first robot pose.

Note that in this smoothing framework, Eq. 2 requires us to re-compute the likelihoods $p\left(I_t\,|\,x_t,\theta\right)$ for *all* past images at every time step and every optimization iteration. This is very expensive due to many pixel-based operations in the likelihood computation. In the following sections, we describe the measurement model $p\left(I_t\,|\,x_t,\theta\right)$, and an efficient approximation of the past likelihood $p\left(I_t\,|\,x_t,\theta\right), t=1..T-1$, based on the current best estimates of $x_{1..T-1}$ and θ, that makes full trajectory and map smoothing tractable without "baking in" linearization errors.

4.1 The Measurement Model $p\left(I_t\,|\,x_t,\theta\right)$

We define the measurement likelihood directly on the image gradients, in terms of the projected edges predicted by the robot pose and tree parameters. We model trees as vertical cylinders, which we predict to produce strong image edges. Similarly to Dellaert and Thrope,[15] we directly define the energy of the measurement likelihood as the negative absolute value of the summed image gradient strength along the projected left Π^L_{tj} and right Π^R_{tj} image edges of the cylindrical tree trunk,

$$e_{I_t}\left(x_t,\theta_j\right)\triangleq\alpha\left(\sum_{u\in\Pi^L_{tj}}\nabla I_t^-\left(u\right)-\sum_{u\in\Pi^R_{tj}}\nabla I_t^+\left(u\right)\right). \tag{3}$$

Here, $\nabla I_t^-=\min\left(0,\nabla I_t\right)$ is the image of negative gradients, $\nabla I_t^+=max\left(0,\nabla I_t\right)$ is the image of positive gradients, and α is a weight, which we choose empirically to be 1000 in our experiments. After splitting the image into these positive and negative gradients, we separately convolve each gradient image with a Gaussian blurring kernel (in our experiments with $\sigma=3$ pixels) to increase the basin of attraction during optimization. The measurement likelihood is then

$$p\left(I_t\,|\,x_t,\theta\right)\propto\prod_j\exp\left(-e_{I_t}\left(x_t,\theta_j\right)\right). \tag{4}$$

4.2 Image Measurement Summarization

To perform the full trajectory and map smoothing in Eq. 2, we must evaluate the likelihoods $p\left(I_t\,|\,x_t,\theta\right)$ for all $t=1..T-1$ on the full trajectory and map given *all past* images. This is too expensive to compute regularly during incremental mapping as it is the product of the measurement likelihoods on all previous frames, which involve computations with many pixels.

Instead, we approximate each past likelihood in Eq. 2 as a product of *approximate summary factors* $\tilde{L}_{tj}\left(x_t,\theta_j\right)\approx p\left(I_t\,|\,x_t,\theta_j\right)$ for each tree in each frame. The summary factors $\tilde{L}_{tj}\left(x_t,\theta_j\right)$ are good approximations of the original measurement likelihoods in Eq 4 but are much cheaper to evaluate. We do this by developing an approximate energy function $\tilde{e}_I\left(x_t,\theta_j\right)$ that is cheap to compute but is still a good approximation, near the energy minimum, of the original highly non-linear energy function, i.e. $\tilde{L}_{tj}\left(x_t,\theta_j\right)=\exp\left(-\tilde{e}_I\left(x_t,\theta_j\right)\right)$.

To make this approximation, we "split" the true likelihood function into a part depending on the image, which is expensive to compute but easy to approximate, and a part involving a geometric transform and projection function, which is cheap to compute but highly non-linear. To do this, we note that because the true measurement likelihood depends only

[†] This Markovian assumption is actually an approximation since the on-board pose filter maintains higher-order state information such as velocities and angular rates, which we do not marginalize out. For the purpose of this paper though we assume this discrepancy can be modeled by zero-mean Gaussian noise on the odometry, but note that the full state could easily be approximated in this framework given appropriate control and sensor models.

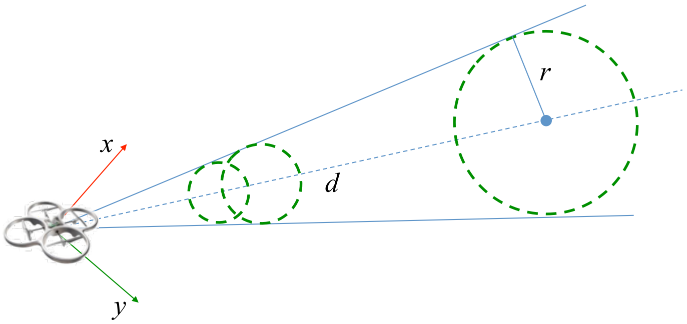


Figure 3: The relative parameters for a *summary factor*. The robot is viewing in its x direction (red). With fixed ratio r/d, the tree could be anywhere along the bearing dotted line.

on the projected tree edges in the image Π_{tj}, it varies only with the *relative* rotation of the tree to the robot, and additionally the size/distance ratio capturing that well-known ambiguity of monocular image (Fig. 3). Hence, we reparameterize the image likelihood energy function in Eq. 3 in terms of these "relative tree parameters" $\theta_{tj}^r \in \mathbb{SO}(3) \times \mathbb{R}^+$,

$$e_I(x_t, \theta_j) = e_I^r\left(\theta_{tj}^r(x_t, \theta_j)\right), \tag{5}$$

where $\theta_{tj}^r : \mathbb{SE}(3) \times \mathbb{R}^3 \to \mathbb{SO}(3) \times \mathbb{R}^+$ is the function converting a robot pose and tree into the corresponding set of relative tree parameters. This function is highly non-linear but very cheap to compute.

More specifically, the relative tree parameterization is $\theta_{tj}^r = \left(R_{tj}^r, \frac{r}{d}\right) \in \mathbb{SO}(3) \times \mathbb{R}^+$, where R_{tj}^r is the relative rotation of the tree to the robot and $\frac{r}{d}$ is the size/distance ratio. The relative distance is $d = \sqrt{\alpha^2 + \beta^2}$, where

$$\begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \theta_j(x) - x_t(x) \\ \theta_j(y) - x_t(y) \end{bmatrix}. \tag{6}$$

The relative rotation is

$$R_{tj}^r = x_t(R)^{-1} R_j, \tag{7}$$

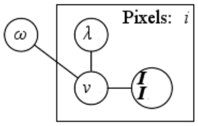
in which the non-relative tree rotation R_j is defined by a coordinate system fixed to the tree, whose x-axis points away from the robot's position in the global x-y plane and whose z-axis points along the world z-axis, and is thus

$$R_j = \begin{bmatrix} \alpha/d & -\beta/d & 0 \\ \beta/d & \alpha/d & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{8}$$

While the image likelihood is expensive to evaluate (due to pixel operations), its log-likelihood is very well-approximated by a quadratic function of these relative tree parameters, and this quadratic approximation need only be computed once if the position *in each image* of the tree edges does not change much during optimization. To ensure a good and unbiased approximation, we non-linearly optimize the relative tree parameters to maximize the true measurement likelihood before computing and replacing the likelihood with its quadratic approximation. Given the current best estimate $(\mathring{x}_t, \mathring{\theta}_j)$ and its relative tree parameters $\mathring{\theta}_{tj}^r = \theta_{tj}^r(\mathring{x}_t, \mathring{\theta}_j)$, we approximate the energy function as:

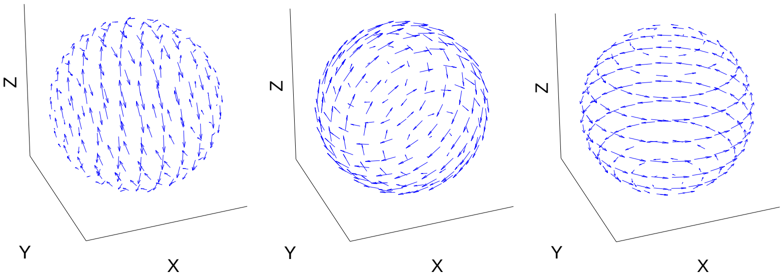
$$e_I(x_t, \theta_j) = e_I^r\left(\theta_{tj}^r(x_t, \theta_j)\right) \approx \mathring{e}_{tj}^r + f_{tj}\delta\theta_{tj}^r + (\delta\theta_{tj}^r)^T F_{tj}\delta\theta_{tj}^r, \tag{9}$$

where $\mathring{e}_{tj}^r = e_I^r\left(\mathring{\theta}_{tj}^r\right) = e_I(\mathring{x}_t, \mathring{\theta}_j)$ is the energy at the approximation center, $f_{tj} = \partial e_I^r\left(\theta_{tj}^r\right)/\partial\theta_{tj}^r$ is the 4×1 Jacobian, $F_{tj} = \partial^2 e_I^r\left(\theta_{tj}^r\right)/\partial\theta_{tj}^{r2}$ is the 4×4 Hessian matrix of the energy function at $\mathring{\theta}_{tj}^r$, and $\delta\theta_{tj}^r = \theta_{tj}^r(x_t, \theta_j) \ominus \mathring{\theta}_{tj}^r$ is the change in the relative tree parameters from the approximation center. The quadratic approximation comprised of $\mathring{e}_{tj}$, f_{tj}, and F_{tj}, is only computed once, while $\delta\theta_{tj}^r$ is recomputed as the optimization and incremental map-building progresses.



$$\lambda_{it} = \begin{cases} 1 & \text{if } i^{th} \text{ pixel} \\ & \text{is background} \\ 0 & \text{if } i^{th} \text{ pixel} \\ & \text{is foreground} \end{cases}$$

Figure 4: Bayes Net depicting our likelihood model of image gradient measurements.



(a) Rotation about $+X$, $\omega = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^\mathsf{T}$.

(b) Rotation about $+Y$, $\omega = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^\mathsf{T}$.

(c) Rotation about $+Z$, $\omega = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^\mathsf{T}$.

Figure 5: The three basis flows of the rotational flow subspace, for rotation about canonical camera axes for a spherical imaging surface. Each basis flow is one column of the flow matrix V.

In practice, the effect of this approximation is that during optimization, the range to trees from the robot trajectory can change and become more certain dramatically as a widening baseline improves the range information. Also, the trajectory can bend and rotate as needed as the trajectory is refined, without introducing significant approximation errors into the summary factors. This makes full trajectory and map smoothing tractable without "baking in" linearization errors.

5. SALIENCY FROM IMMERSIVE OPTICAL FLOW

To reduce the amount of computation, we use a bio-inspired motion saliency measure to focus on nearby trees and ignore innocuous objects that are far away from the robot. We identify *motion-salient* image regions nearby to the camera by their motion parallax. To do this, we simultaneously estimate the camera rotation and segment image regions containing nearby structure from those with distant structure. To avoid the aperture problem of optical flow, instead of directly estimating image motion, we instead directly infer the camera rotation and segmentation from spatial image gradients. This is a modification of the method presented by Roberts *et al.*[24]

Our goal is to label each pixel with the probability that it belongs to the background, $p\left(\lambda_{it} \,|\, I_{t-1}, I_t\right)$, given image intensity measurements of the previous I_{t-1} and current frame I_t. The foreground/background label λ_{it} is a binary indicator variable. Observing the previous and current images I_{t-1} and I_t, we infer these labels in the generative model in Fig. 4, where the foreground/background labels are drawn I.I.D. from a binomial distribution, image velocities are consistent with rotational motion if part of the background, and the image velocities are consistent with constant brightness of image locations corresponding between the previous and current images.

Exact inference of the labels would be intractable, requiring summing over all combinations of the label λ_{it} for every pixel, so instead we apply Expectation Maximization, which alternates between inferring the label probabilities $p\left(\lambda_{it} \,|\, \hat{\omega}_t, I_{t-1}, I_t\right)$ given an estimate of the angular velocity $\hat{\omega}_t$, and estimating the angular velocity by maximizing its expected log-likelihood with respect to the current estimate of the label probabilities. EM avoids the combinatorial search over the labels of all pixels and is guaranteed to converge to a local minimum.[25]

5.1 Inferring the Labels $p\left(\lambda_{it} \,|\, \hat{\omega}_t, I_{t-1}, I_t\right)$ in the E-Step

Using Bayes' law, we can rewrite the label probabilities as

$$p\left(\lambda_{it}{=}1 \,|\, \hat{\omega}_t, I_{t-1}, I_t\right) = \frac{p\left(I_{t-1}, I_t \,|\, \lambda_{it}, \hat{\omega}_t\right) p\left(\lambda_{it}\right)|_{\lambda_{it}=1}}{\sum_{\lambda_{it}=\{0,1\}} p\left(I_{t-1}, I_t \,|\, \lambda_{it}, \hat{\omega}_t\right) p\left(\lambda_{it}\right)}, \tag{10}$$

where $p\left(I_{t-1}, I_t \,|\, \lambda_{it}, \hat{\omega}_t\right)$ is an image likelihood model measuring how consistent a pair of frames is with a given foreground/background labeling and camera rotational velocity, as described above. Also as outlined above, we factor this into a probabilistic brightness constancy constraint and a rotational image motion model at each pixel,

$$p\left(I_{t-1}, I_t \,|\, \lambda_{it}, \hat{\omega}_t\right) = \prod_i \int_{v_{it}} p\left(I_{t-1}, I_t \,|\, v_{it}\right) p\left(v_{it} \,|\, \lambda_{it}, \omega_t\right). \tag{11}$$

The probabilistic brightness constancy constraint[‡] is

$$p\left(I_{t-1}, I_t \,|\, v_{it}\right) = \mathcal{N}\left(I_{t-1}\left(u_i + v_{it}\right) - I_t\left(u_i\right); \mathbf{0}, \sigma_I\right), \tag{12}$$

where $u_i \in \mathbb{R}^2$ is the image location of the i^{th} pixel, $v_{it} \in \mathbb{R}^2$ is the image velocity of the i^{th} pixel at time t, and σ_I is a small amount of Gaussian noise on the pixel intensity. Like the standard brightness constancy constraint from optical flow estimation,[26] when the image is warped between frames due to camera rotation, the brightness of any image location should not change as it moves, except with a small amount of Gaussian noise.

The rotational image motion model is

$$p\left(v_{it} \,|\, \lambda_{it}, \omega_t\right) = \begin{cases} \mathcal{N}\left(v_{it}; V_i \omega_t, \Sigma_b\right), & \text{if } \lambda_{ti} = 1 \\ \mathcal{N}\left(v_{it}; \mathbf{0}, \Sigma_f\right), & \text{if } \lambda_{ti} = 0, \end{cases} \tag{13}$$

$\Sigma_b \in \mathbb{R}^{2\times 2}$ is the covariance of the small noise expected for background rotational flow, and Σ_f is the covariance of the large noise expected for foreground flow including motion parallax. $V_i \in \mathbb{R}^{2\times 3}$ is a matrix mapping angular velocity to the optical flow vector at the i^{th} image location. For the perspective camera we use in our experiments, we can directly calculate the flow matrix, given by,[27] as

$$V_i = \frac{1}{f} \left[\begin{array}{ccc} x_i y_i & -f^2 - x_i^2 & f y_i \\ f^2 + y_i^2 & -x_i y_i & -f x_i \end{array} \right], \tag{14}$$

where x_i and y_i are the horizontal and vertical image location and f is the camera focal length. It is also straightforward to learn this "flow matrix" V as well as the noise parameters Σ_b and Σ_f from data, for imaging systems with near-arbitrary optics, as discussed in.[24]

The linearity of optical flow with respect to rotational velocity is key to efficient inference. Fig. 5 shows the three basis flows of this linearity for a spherical imaging surface; they combine linearly to produce any observable rotational optical flow field, and the coefficients of their linear combination are the components of the angular velocity vector ω.

In order to marginalize out the unknown image velocity v_{it} in Eq 11 in closed-form, we approximate the probabilistic brightness constancy likelihood as a Gaussian about an optical flow estimate $\mathring{v}_t = V \mathring{\omega}_t$, where $\mathring{\omega}_t \leftarrow \hat{\omega}_t$ is the last estimate of the angular velocity, linearizing the image at each image location,

$$p\left(I_{t-1}, I_t \,|\, v_{it}\right) \approx \mathcal{N}\left(I_{it}\left(u_i + \mathring{v}_{it}\right) + \mathring{I}_{it} \delta v_{it}; I_{i,t-1}\left(u_i\right), \sigma_I\right), \tag{15}$$

where the shorthand $\mathring{I}_{it} = \nabla I_t \left(u_i + \mathring{v}_{ti}\right)$ is a 1×2 horizontal vector of the spatial image gradients evaluated at $\mathring{v}_{it}$, and $\delta v_{it} = v_{it} - \mathring{v}_{it}$ is the offset from the linearization point.

5.2 Estimating the Rotational Velocity ω in the M-Step

In the M-step we update the rotation estimate by minimizing the negative expected log-likelihood of the rotation given the labels and images

$$\hat{\omega}_t \leftarrow \arg\min_{\omega_t} \left\langle \log p\left(\omega_t \,|\, \lambda_{it}, I_{t-1}, I_t\right) \right\rangle_{\lambda_{it}} = \arg\min_{\omega_t} \sum_{\lambda_{it} \in \{0,1\}} p\left(\lambda_{it} \,|\, \hat{\omega}_t, I_{t-1}, I_t\right) \log p\left(\omega_t \,|\, \lambda_{it}, I_{t-1}, I_t\right). \tag{16}$$

[‡] We use the notation $\mathcal{N}(x; \mu, \Sigma)$ for the PDF of a Gaussian with mean μ and covariance Σ, evaluated for the expression x.

Subtly but importantly, $\hat{\omega}_t$ on the right-hand-side is the rotation estimate from the previous EM iteration, while ω_t on the right-hand-side is the update being estimated at each iteration. Using the same methods as in Section 5.1, we write the rotation likelihood as

$$p\left(\omega_t \,|\, \lambda_{it}, I_{t-1}, I_t\right) \propto p\left(I_{t-1}, I_t \,|\, \lambda_{it}, \omega_t\right) p\left(\omega_t\right) = \prod_i \int_{u_{it}} p\left(I_{t-1}, I_t \,|\, v_{it}\right) p\left(v_{it} \,|\, \lambda_{it}, \omega_t\right) p\left(\omega_t\right). \tag{17}$$

Minimizing Eq 16 is a non-linear least-squares problem, so like above in Eq 15, we linearize the image about the current estimate of the rotational velocity $\mathring{\omega}_t$ (approximating the image likelihood $p\left(I_{t-1}, I_t \,|\, u_{it}\right)$ as a Gaussian), to give an approximate linear least-squares problem. This comprises one iteration of a Gauss-Newton method. To reduce the total number of iterations, we perform only a single Gauss-Newton update in each M-Step, only relinearizing after updating the foreground/background labels λ_{it}.

6. DETECTING TREES IN SALIENT REGIONS

Given the probabilistic saliency map computed in the previous Section 5, our detector selects pairs of candidate lines in the salient image regions that have high probability of being a left border and a right border of a tree. Beside picking up trees only in salient regions, we further reduce the amount of computation by only looking for trees in unexplored areas and ignoring regions that might contain previously detected trees. By focusing on unpopulated image regions, we can reduce the chance of detecting and adding similar objects into the system, which might both corrupt the map and increase the amount of computation unnecessarily.

Knowing that tree trunks are mostly vertical in the image, we generate projections of 3D vertical lines in the image using the predicted robot orientation obtained from the mapper, then compute the probabilities of these lines to be either left or right borders of new salient trees. These probabilities are computed from the gradient images $\nabla I_t^-, \nabla I_t^+$ as in Eq 3, the saliency map of foreground/background probability from Section 5, and the non-inhibited regions projected from our current best estimate of existing trees θ.

Specifically, the probability of a line to be a left border of a new salient tree is computed as:

$$p\left(\text{new\_left\_border} \,|\, \nabla I_t^-, \langle\lambda_{it}\rangle, \theta\right) = p\left(\text{left} \,|\, \nabla I_t^-\right) p\left(\text{foreground} \,|\, \langle\lambda_t\rangle\right) p\left(\text{non\_inhib} \,|\, \theta\right), \tag{18}$$

and the probability of a line to be a right border of a new salient tree is also computed in a similar way using the positive image gradient ∇I_t^+.

We then apply non-maxima suppression on these probabilities and select non-overlapping pairs of lines with probabilities higher than a fixed threshold, and instantiate new trees from them. This threshold could also be learned from the data, but we choose it empirically as 0.75 for our experiments.

We compute the probabilities $p\left(\text{left} \,|\, \nabla I_t^-\right)$ and $p\left(\text{foreground} \,|\, \langle\lambda_t\rangle\right)$ using logistic function:

$$p(\text{left} \,|\, \nabla I_t^-) = Q\left(\frac{1}{N}\sum_{i \in l} \nabla I_t^-(i)\,;\, a_{left}, b_{left}\right), \tag{19}$$

$$p\left(\text{foreground} \,|\, \langle\lambda_t\rangle\right) = Q\left(\frac{1}{N}\sum_{i \in l}\left(1 - \langle\lambda_{it}\rangle\right); a_s, b_s\right), \tag{20}$$

where N is the number of pixels along the line l, and Q is the logistic function defined as:

$$Q(x; a, b) = \frac{1}{1 + e^{ax+b}},$$

where a and b are parameters. The logistic function can intuitively be seen as a "soft threshold" that directly approximates the conditional density of a binary variable given a continuous one. In our experiments we choose the parameters a and b by inspection of the statistics of the data, but they can also be easily learned from supervised data.

Finally, the probability of a line not being inhibited by existing trees is computed as:

$$p\left(\text{non\_inhib} \,|\, \theta\right) = \max\left(1, \sum_{\theta_i} \exp \frac{-1}{2} \sigma_R^{-2} d_{\theta_i}^2\right), \tag{21}$$

where d_{θ_i} is the distance from the line to the i^{th} tree, and σ_R is a parameter that controls the extent of inhibition of a tree.

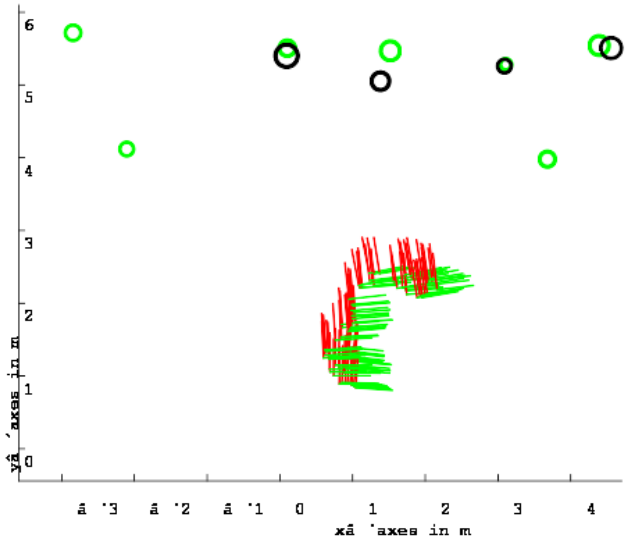


Figure 6: Comparison of ground truth tree positions (green circles) and solution of our system (red circles), along with the robot trajectory. Maps were hand-aligned to the ground truth.

7. EXPERIMENTS AND RESULTS

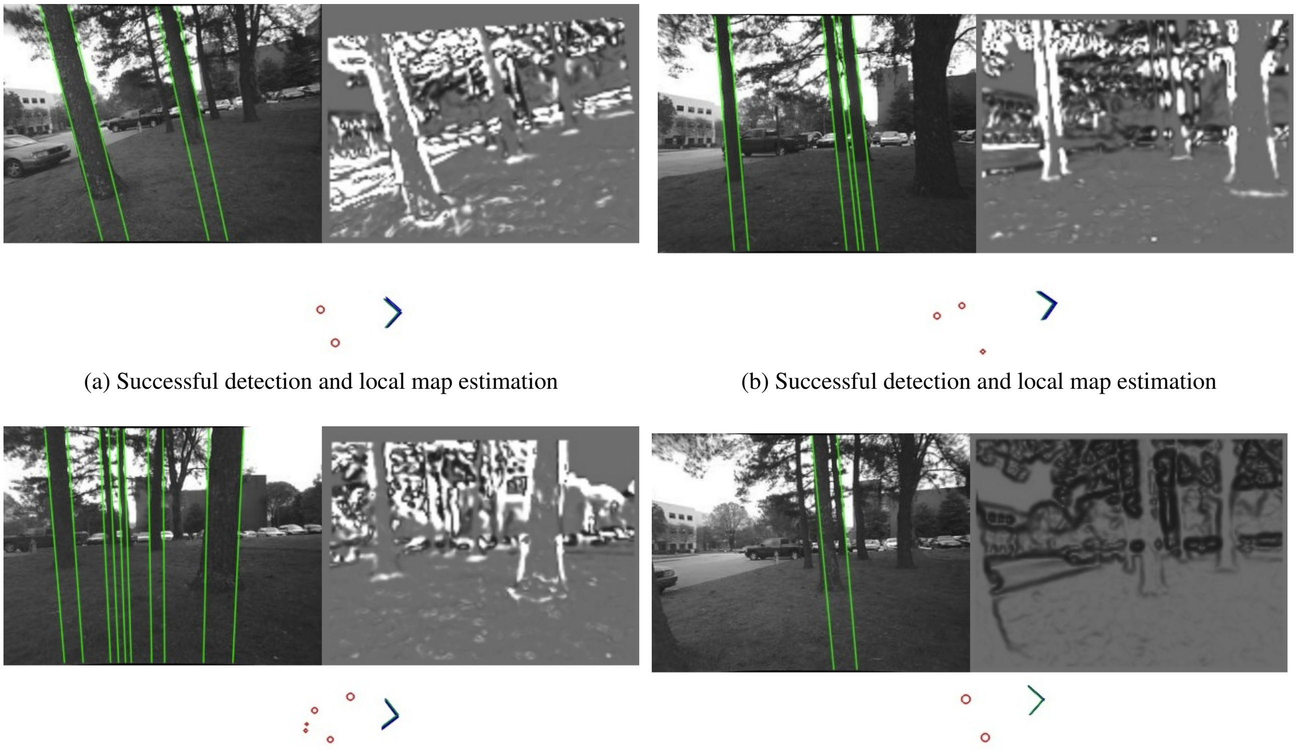
For the evaluation of our approach, we manually flew the commercially-available and inexpensive AR.Drone through a small forest as can be seen in Fig. 1. The AR.Drone is a small, lightweight quadrotor equipped with a front-facing camera and an on-board pose filter and controller that integrates information from an accelerometer, gyroscope, downward-ranging sonar, and downward-facing camera. We stream wirelessly from the quadrotor 320×240 pixel grayscale frames from the front facing camera at 10 Hz as well as pose filter and IMU sensor data to a laptop, where they are stored for offline processing. The IMU data comprises acceleration as well as rotation speed data in all three respective axes, while the pose filter data includes 6-DOF pose. The angular rate data from the 3-DOF gyroscope was used to obtain initial rotation estimates for the motion saliency estimation in Section 5. The offline processing of the datasets was done on a 2.2 GHz Core i7 laptop. Our research implementation runs at approximately one third of real-time, but the implementation has known inefficiencies and can be made significantly faster.

In addition to the methods described above, we use a simple heuristic to stop estimation of trees when their image measurements yield energy functions with negative curvature, meaning that they have diverged from energy minima. This typically occurs when a tree begins to become occluded or occludes another tree. After the trees are unoccluded, our method typically redetects them but we do not attempt to re-associate them with existing trees. This causes some duplicate trees in our maps, and could make a motion planner overly cautious but would not cause collisions. Our system produces a small local map around the robot in a global coordinate frame, with the first robot pose fixed to the origin.

7.1 Local Tree Maps

As described in Section 1, the goal is that our maps contain low-dimensional geometric information immediately useful to obstacle avoidance and path planning. Therefore, we build local maps of the trees around the robot, which could be used as input to a planner or controller. Figure 6 shows a comparison of one of these local maps with ground-truth locations for the trees in the vicinity of the robot. Figure 7 shows typical success and failure cases for our vision system.

We also claimed that motion saliency segments the nearby objects, which are important for local motion planning, from the distant ones, which are not. Fig. 8 shows how the computation of motion saliency in the image aids the detection and tracking of nearby trees. In the saliency images, white color depicts closeby motion salient regions, whereas black shows distant areas which exhibit no or only slow motion. Comparing the original image from the AR.Drone camera with the respective saliency image clearly shows that motion saliency is a good predictor for closeby trees.



(a) Successful detection and local map estimation

(b) Successful detection and local map estimation

(c) Successful detection and local map estimation

(d) Detection failed for several trees due to lack of sufficient motion parallax for saliency estimation. Tracking also failed for a nearby tree due to revealing of an occluded faraway tree.

Figure 7: Success and failure cases in detecting salient trees and building local maps.

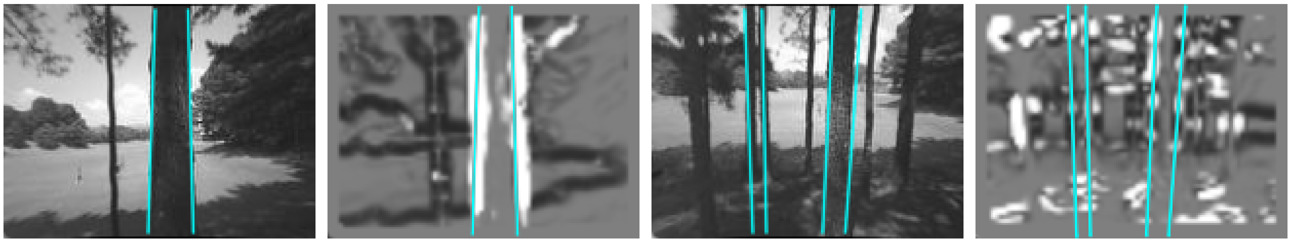


Figure 8: Images 1 and 3 from the left show the greyscale image obtained from the AR.Drone camera, whereas images 2 and 4 show respective saliency field. Highly motion salient regions are colored white, whereas distant areas with no or only slow motion are depicted black. The cyan lines in all images depict the newly-detected salient trees.

8. CONCLUSION AND FUTURE WORK

We have presented a system that makes significant progress towards the goal of fast, vision-based autonomous flight, localization, and map building to support local planning and control in unstructured outdoor environments. We estimate low-dimensional geometric information about trees viewed by a quadrotor flying through a forest, which do not exhibit point features suitable for building point clouds. We use motion saliency and developed nonlinear image summary factors to keep computational complexity down while mapping relevant objects and maintaining accuracy.

We intend this work as a demonstration of what information can be estimated using only these methods, and to identify where it makes sense to augment these methods with additional information to improve accuracy and consistency. Having better pose estimates would allow us to track through occlusions, and thus our future work is to track a small number of point features to aid in pose estimation.

Additionally, while we currently decouple the image motion reasoning that takes place in computing motion saliency from map inference, we believe that the map accuracy, reliability, as well as computational efficiency, could be improved by integrating the image brightness and motion models discussed in Section 5 with the full joint inference problem. Not only would this improve pose estimates, but image motion information from the tree edges could also inform estimates of the tree positions relative to the robot.

Finally, while we gear our methods towards low computation, as stated previously we perform all processing offline on a laptop. In future work we will address performing these computations entirely on-board the quadrotor.

Acknowledgements

This work is supported by an ARO MURI grant, award number W911NF-11-1-0046.

REFERENCES

1. Grzonka, S., Grisetti, G., and Burgard, W., "Towards a navigation system for autonomous indoor flying," in [*Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*], 2878–2883 (2009).
2. Bachrach, A., He, R., and Roy, N., "Autonomous flight in unknown indoor environments," *International Journal of Micro Air Vehicles* **1**(4), 217–228 (2009).
3. Achtelik, M., Bachrach, A., He, R., Prentice, S., and Roy, N., "Stereo vision and laser odometry for autonomous helicopters in GPS-denied indoor environments," *Unmanned Systems Technology XI. Ed. Grant R. Gerhart, Douglas W. Gage, & Charles M. Shoemaker. Orlando, FL, USA: SPIE* (2009).
4. Scherer, S., Singh, S., Chamberlain, L., and Saripalli, S., "Flying fast and low among obstacles," in [*Robotics and Automation, 2007 IEEE International Conference on*], 2023–2029 (2007).
5. Soundararaj, S., Sujeeth, A., and Saxena, A., "Autonomous indoor helicopter flight using a single onboard camera," in [*Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*], 5307–5314 (2009).
6. Steder, B., Grisetti, G., Stachniss, C., and Burgard, W., "Visual SLAM for flying vehicles," *Robotics, IEEE Transactions on* **24**(5), 1088–1093 (2008).
7. Blösch, M., Weiss, S., Scaramuzza, D., and Siegwart, R., "Vision based mav navigation in unknown and unstructured environments," in [*Robotics and Automation (ICRA), 2010 IEEE International Conference on*], 21–28 (2010).
8. Achtelik, M., Achtelik, M., Weiss, S., and Siegwart, R., "Onboard imu and monocular vision based control for mavs in unknown in- and outdoor environments," in [*Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*], (2011).
9. Weiss, S., Achtelik, M., Kneip, L., Scaramuzza, D., and Siegwart, R., "Intuitive 3d maps for mav terrain exploration and obstacle avoidance," *Journal of Intelligent and Robotic Systems* **61**, 473–493 (2011).
10. Klein, G. and Murray, D., "Parallel tracking and mapping for small AR workspaces," in [*IEEE and ACM Intl. Sym. on Mixed and Augmented Reality (ISMAR)*], 225–234 (Nov 2007).
11. Langelaan, J. and Rock, S., "Towards autonomous uav flight in forests," in [*Proc. of AIAA Guidance, Navigation and Control Conference*], (2005).
12. Kemp, C., *Visual control of a miniature quad-rotor helicopter*, PhD thesis, Churchill College University of Cambridge (2005).

13. Teulière, C., Eck, L., Marchand, E., and Guénard, N., "3D model-based tracking for UAV position control," in [*Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*], 1084–1089 (2010).
14. Canny, J., "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Machine Intell.* **8**, 679–698 (November 1986).
15. Dellaert, F. and Thorpe, C., "Robust car tracking using Kalman filtering and Bayesian templates," in [*Intl. Soc. Opt. Eng. (SPIE)*], **3207** (October 1997).
16. Kollnig, H. and Nagel, H. H., "3D pose estimation by directly matching polyhedral models to gray value gradients," *International Journal of Computer Vision* **23**(3), 283–302 (1997).
17. Marchand, E., Bouthemy, P., and Chaumette, F., "A 2D-3D model-based approach to real-time visual tracking," *Image and Vision Computing* **19**, 941–955 (Nov. 2001).
18. Gibson, J. J., "Visually controlled locomotion and visual orientation in animals," *British journal of psychology* **49**, 182–194 (1958).
19. Wolbers, T., Hegarty, M., Büchel, C., and Loomis, J., "Spatial updating: how the brain keeps track of changing object locations during observer motion," *Nature Neuroscience* **11**(10), 1223–1230 (2008).
20. Gould, S., Arfvidsson, J., Kaehler, A., Sapp, B., Meissner, M., Bradski, G., Baumstarck, P., Chung, S., and Ng, A. Y., "Peripheral-foveal vision for real-time object recognition and tracking in video," in [*Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI-07)*], (2007).
21. Itti, L., Koch, C., and Niebur, E., "A model of saliency-based visual attention for rapid scene analysis," *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **20**(11), 1254–1259 (1998).
22. Itti, L. and Koch, C., "Computational modelling of visual attention," *Nature reviews neuroscience* **2**(3), 194–203 (2001).
23. Dellaert, F. and Kaess, M., "Square Root SAM: Simultaneous localization and mapping via square root information smoothing," *Intl. J. of Robotics Research* **25**, 1181–1203 (Dec 2006).
24. Roberts, R., Potthast, C., and Dellaert, F., "Learning general optical flow subspaces for egomotion estimation and detection of motion anomalies," in [*IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*], (2009).
25. Dempster, A., Laird, N., and Rubin, D., "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society, Series B* **39**(1), 1–38 (1977).
26. Horn, B. and Schunck, B., "Determining Optical Flow," *Artificial Intelligence* **17**(1-3), 185–203 (1981).
27. Heeger, D. and Jepson, A., "Visual perception of three-dimensional motion," *Neural Computation* **2**, 127–137 (1990).