Adaptive Pose Estimation for Different Corresponding Entities

Bodo Rosenhahn, Gerald Sommer

Institut für Informatik und Praktische Mathematik
Christian-Albrechts-Universität zu Kiel
Preußerstrasse 1-9, 24105 Kiel, Germany
{bro,gs}@ks.informatik.uni-kiel.de

Abstract. This paper concerns the 2D-3D pose estimation problem for different corresponding entities. Many articles concentrate on specific types of correspondences (mostly point, rarely line correspondences). Instead, in this work we are interested to relate the following image and model types simultaneously: 2D point/3D point, 2D line/3D point, 2D line/3D line, 2D conic/3D circle, 2D circle/3D sphere. Furthermore, to handle also articulated objects, we describe kinematic chains in this context in a similar manner. We further discuss the use of weighted constraint equations, and different numerical solution approaches.

1 Introduction

In this work we derive a solution approach for simultaneous 2D-3D pose estimation from different corresponding entities. Pose estimation itself is a basic visual task [3] and the first solution approaches were presented in the early eighties [7]. Monocular pose estimation means to relate the position of a 3D object to a reference camera coordinate system [14, 10][1]. Nearly all papers concentrate on one specific type of correspondences. But many situations are conceivable in which a system has to gather information from different hints or has to consider different reliabilities of measurements. This is the main aspect of this work: We describe a scenario for adaptive pose estimation of simultaneously used different entities, without loosing linearity, good conditioned equations and real-time capability.

The scenario of pose estimation

In the scenario of figure 1 we describe the following situation: We assume points, lines, spheres, circles or kinematic chain segments of an 3D object or reference model. Further, we extract corresponding 2D features in an image of a calibrated camera. The aim is to find the rotation $\boldsymbol{R}$ and translation $\boldsymbol{t}$ of the object, which lead to the best fit of the reference model with the actual projective reconstructed entities. This means, an image point is reconstructed to a *projection ray*, or an image line is reconstructed to a *projection plane*. Then constraints are build in the 3D space to compare the model features with the reconstructed image features.

[1] Many other scientists also concern this problem in several variations, but we can not quote them due to the space limits.

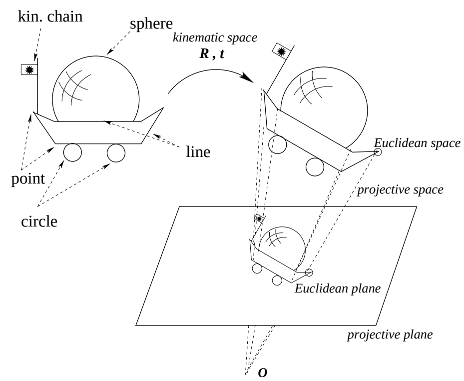

Fig. 1. The scenario. The solid lines describe the assumptions: the camera model, the model of the object (consisting of points, lines, circles, spheres and kinematic chains) and corresponding extracted entities on the image plane. The dashed lines describe the pose of the model, which leads to the best fit of the object with the actual extracted entities.

2 Geometric Algebras

We use geometric algebras to formalize the geometric scenario and the pose estimation process. The advantage of this language is its dense symbolic representations of higher order entities with linear operations acting on those. In this contribution we will not give a detailed introduction in geometric algebras. This can be found in [13]. The main idea of geometric algebras $\mathcal{G}$ is to define a product on basis vectors, which extends the linear vector space V of dimension n to a linear space of dimension 2^n.The elements are so-called multivectors as higher order algebraic entities in comparison to vectors of a vector space as first order entities. A geometric algebra is denoted as $\mathcal{G}_{p,q}$ with $n = p+q$. Here p and q indicate the numbers of basis vectors which square to $+1$ and -1, respectively. The product defining a geometric algebra is called *geometric product* and is denoted as $\boldsymbol{uv}$ for two multivectors $\boldsymbol{u}$ and $\boldsymbol{v}$. Operations between multivectors can be expressed by special products, called *inner* $\cdot$, *outer* $\wedge$, *commutator* $\underline{\times}$ and *anticommutator* $\overline{\times}$ product. The most powerful and only recently introduced algebra is the conformal geometric algebra $\mathcal{G}_{4,1}$ (ConfGA) [8]. Because it is suited to describe conformal geometry, it contains spheres as entities and a rich set of geometric manipulations. The point at infinity, $\mathbf{e}$, and the origin, $\mathbf{e}_0$, are special elements and define a null space in the conformal geometric algebra.

Rigid transformations in ConfGA

Rotations are represented by rotors, $\boldsymbol{R} = \exp\left(\frac{\theta}{2}\boldsymbol{l}\right)$. The components of the rotor $\boldsymbol{R}$ are the unit bivector $\boldsymbol{l}$ which represents the dual of the rotation axis, and the angle θ which represents the amount of the rotation. The rotation of an entity can be performed by its spinor product $\underline{\boldsymbol{X}}' = \boldsymbol{R}\underline{\boldsymbol{X}}\widetilde{\boldsymbol{R}}$. A translation can be expressed by a translator, $\boldsymbol{T} = (1 + \frac{\mathbf{e}\boldsymbol{t}}{2}) = \exp\left(\frac{\mathbf{e}\boldsymbol{t}}{2}\right)$. To estimate the rigid body motion (containing a rotor $\boldsymbol{R}$ and translation vector $\boldsymbol{t}$), we follow e.g. [9]: A rigid body motion can be expressed by a rotation about a line in space. This results from the fact that for every $g \in SE(3)$ exists a $\xi \in se(3)$ and a $\theta \in \mathbb{R}$ such that $g = \exp(\xi\theta)$. The element ξ is also called a *twist*. The motor

$\boldsymbol{M}$ describing a twist transformation has the general form $\boldsymbol{M} = \boldsymbol{T}\boldsymbol{R}\widetilde{\boldsymbol{T}}$, denoting the inverse translation, rotation and back translation, respectively. But whereas in Euclidean geometry, Lie algebras and Lie groups are only applied on point concepts, the motors and twists can also be applied on other entities, like lines, planes, circles, spheres, etc.

Constraint equations for pose estimation

Now we express the 2D-3D pose estimation problem, *a transformed object entity has to lie on a spatial entity, projective reconstructed from an image entity.* Let $\underline{\boldsymbol{X}}$ be an object point and $\underline{\boldsymbol{L}}$ be an object line, given in ConfGA. The (unknown) transformations of the entities can be described as $\boldsymbol{M}\underline{\boldsymbol{X}}\widetilde{\boldsymbol{M}}$ and $\boldsymbol{M}\underline{\boldsymbol{L}}\widetilde{\boldsymbol{M}}$, respectively. Let $\boldsymbol{x}$ be an image point and $\boldsymbol{l}$ be an image line on a projective plane. The projective reconstruction of an image point in ConfGA can be written as $\underline{\boldsymbol{L}}_x = \mathbf{e} \wedge \boldsymbol{o} \wedge \boldsymbol{x}$. The entity $\underline{\boldsymbol{L}}_x$ is a circle, containing the vector $\boldsymbol{o}$ as the optical center of the camera, see e.g. figure 1, the image point $\boldsymbol{x}$ and the vector $\mathbf{e}$ as the point at infinity. This leads to a reconstructed projection ray. Similarly leads $\underline{\boldsymbol{P}}_l = \mathbf{e} \wedge \boldsymbol{o} \wedge \boldsymbol{l}$ to a reconstructed projection plane in ConfGA. Collinearity and coplanarity can be described by the commutator and anticommutator products. Thus, the constraint equations of pose estimation from image points read

$$\underbrace{\underbrace{(\boldsymbol{M} \underbrace{\underline{\boldsymbol{X}}}_{\text{object point}} \widetilde{\boldsymbol{M}})}_{\text{rigid motion of the object point}} \underline{\times} \underbrace{\mathbf{e} \wedge (\boldsymbol{o} \wedge \boldsymbol{x})}_{\substack{\text{projection ray,} \\ \text{reconstructed from the image point}}}}_{\substack{\text{collinearity of the transformed object} \\ \text{point with the reconstructed line}}} = 0.$$

Constraint equations to relate 2D image lines to 3D object points, or 2D image lines to 3D object lines, can be expressed in a similar manner. Note: The constraint equations in the unknown motor $\boldsymbol{M}$ express a distance measure which has to be zero. But in contrast to other approaches, where the minimization of errors has to be computed directly on the geometric transformations [2], in our approach a distance in the Euclidean space constitutes the error measure.

3 Pose estimation with extended object concepts

This section concerns the derivation of constraint equations for kinematic chains, circles and spheres.

Kinematic chains

With *kinematic chains* we mean linked rigid objects which can only change their pose in mutual dependence. Examples are tracked robot arms or human body movements, see e.g. figure 4. So far we have parameterized the 3D pose constraint equations of a rigid object. Assume that a second rigid body is attached to the first one by a joint. The joint can be formalized as an axis of rotation and/or translation in the object frame (*revolute* or *prismatic* joint respectively). Each joint defines a new coordinate system, and the coordinate transformations between joints can be described by suitable motors $\boldsymbol{M}_j$. This means, an entity given in the coordinate system of the jth joint can be translated in an entity of the base coordinate system by transforming the entity with the motors $\boldsymbol{M}_1, \ldots, \boldsymbol{M}_j$. The points attached to the j-th joint are numbered as $\underline{\boldsymbol{X}}_{j,1},\ldots,\underline{\boldsymbol{X}}_{j,i_j}$. The transformation of the points on the j-th joint in terms of the base coordinate system can be formalized as $\underline{\boldsymbol{X}}^0_{j,i_j}=\boldsymbol{M}_1\ldots\boldsymbol{M}_j\underline{\boldsymbol{X}}_{j,i_j}\widetilde{\boldsymbol{M}}_j\ldots\widetilde{\boldsymbol{M}}_1$.

Now we will combine the introduced representation of a kinematic chain with the pose estimation constraints derived in the previous section. The pose of the base corresponds to a motor $\boldsymbol{M}$. The constraint equation for a point at the j-th joint leads to

$$(\boldsymbol{M}(\boldsymbol{M}_1\ldots\boldsymbol{M}_j\underline{\boldsymbol{X}}_{j,i_j}\widetilde{\boldsymbol{M}}_j\ldots\widetilde{\boldsymbol{M}}_1)\widetilde{\boldsymbol{M}})\ \underline{\times}\ \mathbf{e}\wedge(\boldsymbol{o}\wedge\boldsymbol{x}_{j,i_j})=0.$$

Circles and spheres

We now explain, how to build constraint equations for 3D circles and 3D spheres. The key idea is to interpret circles and spheres as virtual kinematic chains: A circle can be described by a twist ξ and a point $\underline{\boldsymbol{X}}_C$ on the circle. Let $\boldsymbol{M}_\phi$ be a motor, describing a general rotation around the twist ξ. Then the circle is simply given by all points which result from the transformation of the point $\underline{\boldsymbol{X}}_C$,

$$\underline{\boldsymbol{X}}^\phi_C=(\boldsymbol{M}_\phi\underline{\boldsymbol{X}}_C\widetilde{\boldsymbol{M}}_\phi)\quad:\quad\phi\in[0\ldots2\pi].$$

We can similarly proceed with spheres, just by rotating a point with two twists and gaining the points on a sphere:

$$\underline{\boldsymbol{X}}^{\phi_1,\phi_2}_S=(\boldsymbol{M}^1_{\phi_1}\boldsymbol{M}^2_{\phi_2}\underline{\boldsymbol{X}}_S\widetilde{\boldsymbol{M}^2}_{\phi_2}\widetilde{\boldsymbol{M}^1}_{\phi_1})\quad:\quad\phi_1,\phi_2\in[0\ldots2\pi].$$

The constraint equations for tangentiality of projection rays to circles or spheres can be summarized as

$$(\boldsymbol{M}(\boldsymbol{M}_\phi\underline{\boldsymbol{X}}_C\widetilde{\boldsymbol{M}}_\phi)\widetilde{\boldsymbol{M}})\underline{\times}\mathbf{e}\wedge(\boldsymbol{o}\wedge\boldsymbol{x})=0,$$
$$(\boldsymbol{M}(\boldsymbol{M}^1_{\phi_1}\boldsymbol{M}^2_{\phi_2}\underline{\boldsymbol{X}}_S\widetilde{\boldsymbol{M}^2}_{\phi_2}\widetilde{\boldsymbol{M}^1}_{\phi_1})\widetilde{\boldsymbol{M}})\underline{\times}\mathbf{e}\wedge(\boldsymbol{o}\wedge\boldsymbol{x})=0.$$

4 Experiments

In this section we will show experimental results of pose estimation.

Solving the constraint equations

In the last sections, several constraint equations to relate object informations to image informations are derived. In these equations, the object, camera and image data are assumed to be known and the motor $\boldsymbol{M}$ expressing the motion is assumed to be unknown. There exist several ways to estimate the motion parameters. In earlier works we concerned this problem and we estimated the motion parameters either on the Lie group $SE(3)$ itself (by using an SVD approach), or by using an extended Kalman filter (EKF) [12]. In [11] we presented a new method, which does not estimate the rigid body motion on the Lie group $SE(3)$, but the parameters which generate their Lie algebra $se(3)$ (*twist approach*), comparable to the ideas, presented in [1, 7]. Note: Though the equations are expressed in a linear manner with respect to the group action, the equations

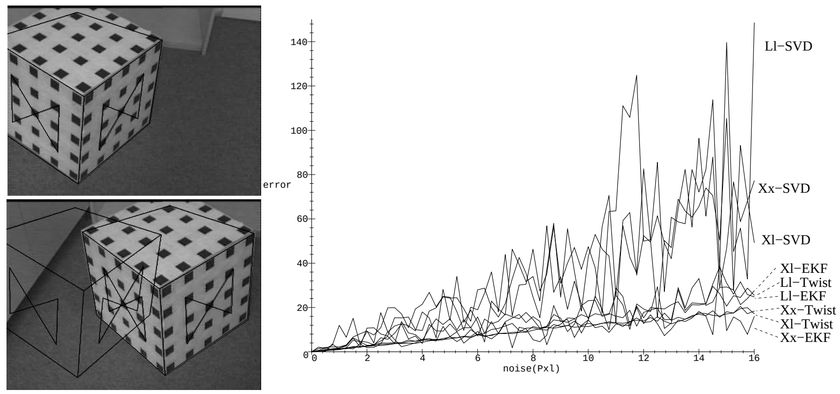

Fig. 2. The scenario of the first experiment. In the first image the calibration is performed and the 3D object model is projected on the image. Then the camera moved and corresponding line segments are extracted. For comparison reasons, the initial pose is overlaid. The diagram shows the performance comparison of different methods in case of noisy data.

in the unknown generators of the group action are non-linear and in the twist approach they will be linearized and iterated.

In our first experiment, we compare the noise sensitivity of these three methods, with respect to the three constraint equations, relating 3D points to 2D points (Xx), 3D points to 2D lines (Xl), or 3D lines to 2D lines (Ll). Therefore we add a Gaussian noise on extracted image points in a virtual scenario (see figure 2). Then we estimate the rigid body motion, and use the translational error between the ground truth and the disturbed values as error measure. The result is depicted in figure 2. It is easy to see, that the results, obtained with the SVD approach are the worst ones. Instead, the Kalman filter and the twist approach have a more stable and comparable error behavior. It is obvious, that the results of the experiments are not much affected by the used constraints themselves. This occurs because we selected certain points directly by hand and derived from these the line subspaces. So the quality of the line subspaces is directly connected to the quality of the point extraction. The result of this investigation is, that for noise corresponding to a distribution function, the Kalman filter or twist approach for pose estimation should be used. There are two main reasons, why we further prefer the twist approach for pose estimation instead of the EKF: Firstly, the Kalman filter is sensitive to outliers (see e.g. figure 4), leading to non-converging results. Secondly, Kalman filters must be designed for special situations or scenarios. So the design of a general Kalman filter, dealing with different entities in a weighted manner is hard to implement. Instead, this can be done very easily in the twist approach since the linearized constraint equations of any entity can just be scaled and put in one system of equations.

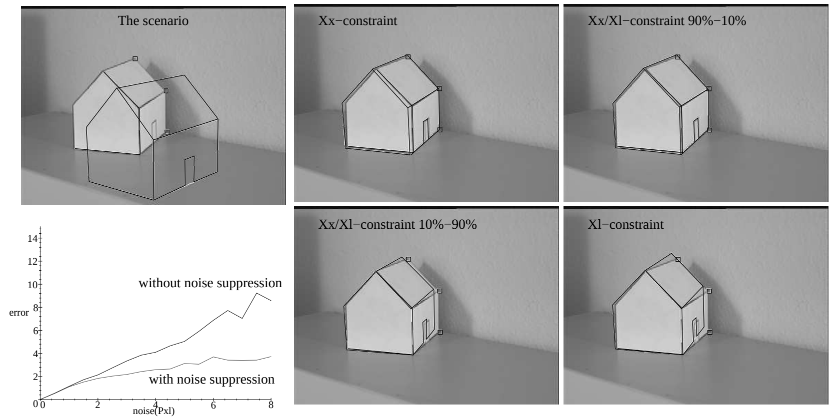

Fig. 3. Different weightings of constraints for pose estimation.

Adaptive use of pose estimation constraints
Image preprocessing algorithms sometimes enable a characterization of the quality of extracted image data. The idea to use these additional information in the context of pose estimation is the following: Every constraint equation describes a distance measure for the involved entities. These constraint equations can be

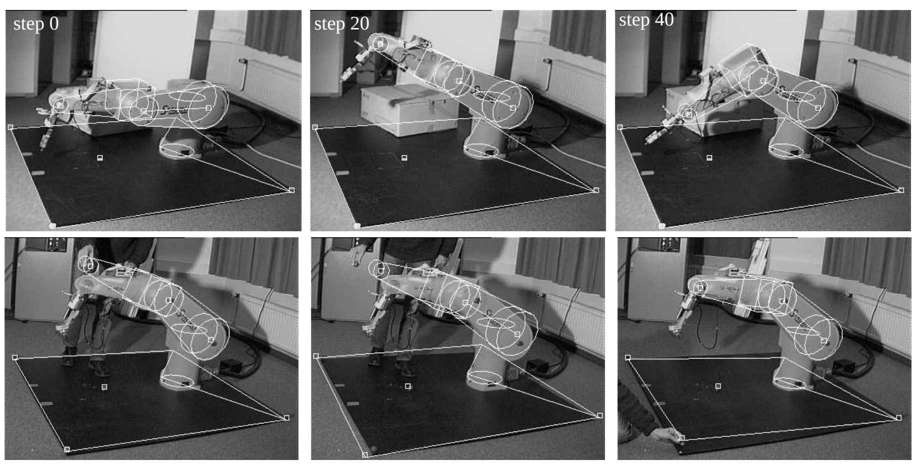

Fig. 4. Images of a tracked robot arm taken from a sequence of 40 images. The second row shows a stability example for disturbed color markers.

scaled by a scalar $\lambda \in \mathbb{R}$ and so it is possible to individually scale the weighting of the equation to the whole equations system. Figure 3 shows an example: On the one hand we have three extracted image points and on the other hand three extracted image lines. We can use both information separately to evaluate the pose of the object. Since we have only few information for each type of correspondences, the object itself is not very well fitted to the image data (see e.g.

the images with the Xx-constraint or Xl-constraint). On the other hand, we can put both constraint equations in one whole system of equations and solve the unknowns by using all image information at once. Furthermore, we are able to choose different weightings of the constraints. The change of the pose estimations is visualized in the other images of figure 3. To address the noise adaptive use of the pose estimation constraints, we add a Gaussian noise on some of the extracted image points. Then we solve the constraint equations with and without weighting the constraints, depending on the noise level. We call this method *noise suppression*. The result is visualized in the diagram of figure 3.

Pose estimation of kinematic chains

In the next experiment (see figure 4), we use as object model a robot arm. We estimate the pose of the robot and the angles of the kinematic chain via tracked point markers. The errors we gain in these experiments are dependent on the calibration quality, lens distortion and accuracy of the point marker detection. They differ around 0.5 till 3 degree. The second row of figure 4 shows images of a second sequence. There we visualize the stability of our algorithm in the context of moved color markers and therewith resulting impossible kinematics: During the tracking, a student moves into the scenario and picks up a color marker and moves it around. The model will not be distorted. Instead, the algorithm leads to a spatial best fit of the model to the extracted image data.

Simultaneous pose estimation with different kinds of correspondences

In the last experiment we use a model which contains a prismatic and revolute joint, 3D points, 3D lines, 3D circles and a 3D sphere. Figure 5 shows some pose estimation results of the object model. Though we measured the size of the model by hand, the pose is accurate and also the joint parameters are good approximated. All information is accumulated in one linear system of equations.

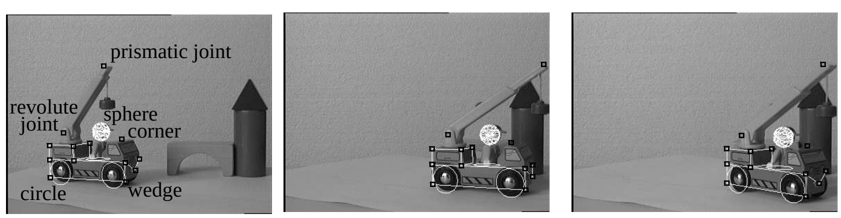

Fig. 5. Pose estimation by using different types of correspondences

This leads to simultaneous solving of the pose parameters by using all features, without following the classical way of point based estimations of subspace concepts in vector space.

5 Discussion

This contribution concerns the simultaneous estimation of 2D-3D pose for different kinds of correspondences. We present a new framework in the language of geometric algebra for pose estimation of object models, which consist of different types of entities, including points, lines, planes, circles, spheres and kinematic chains.

Compared with other algorithms, we are able to use a full perspective camera model in this context and not an orthographic one as e.g. in [1]. We also formulate the equations as differential approximation of the requested group actions and put them in one equations system. This enables an easy use of different entities in the same system. We also discuss different solution approaches for pose estimation and recommend the use of Kalman filters or twists for pose estimation, but not the estimation on the group manifold itself. This result is in contrast to the results presented in [6]. The noise adaptive use of the constraints is also interesting with respect to the design of behavior based or learning robot systems. Only sporadic work concerning this for stable running systems important topic exist so far (e.g. [5]). We implemented the sources in C++ and are able to estimate the motion (and kinematic chain) parameters in real-time with 15 frames per second on a SUN Ultra 10.

References

1. Bregler C. and Malik J. Tracking people with twists and exponential maps. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Santa Barbara, California, pp. 8-15, 1998.
2. Chiuso A. and G. Picci. Visual tracking of points as estimation on the unit sphere. In *The Confluence of Vision and Control*, pp. 90-105, Springer-Verlag, 1998.
3. Grimson W. E. L. Object Recognition by Computer. *The MIT Press, Cambridge, MA*, 1990.
4. Hel-Or Y. and Werman M. Pose estimation by fusing noisy data of different dimensions. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, Vol. 17, No.2, February 1995.
5. Holt J.R. and Netravali A.N. Uniqueness of solutions to structure and motion from combinations of point and line correspondences. *Journal of Visual Communication and Image Representation*, Vol.7:2, pp. 126–136, 1996.
6. Lorusso A., Eggert D.W. and Fisher R.B. A comparison of four algorithms for estimating 3-d rigid transformations. In *British Machine Vision Conference*, Birmingham, pp. 237–246, England, 1995.
7. Lowe D.G. Three-dimensional object recognition from single two-dimensional images. *Artificial Intelligence*, Vol. 31 No. 3, pp. 355-395, 1987.
8. Li H. Generalized homogeneous coordinates for computational geometry. In [13], pp. 27-52, 2001.
9. Murray R.M., Li Z. and Sastry S.S. A Mathematical Introduction to Robotic Manipulation. *CRC Press*, Inc., 1994.
10. Horaud R., Phong T.Q. and Tao P.D. Object pose from 2d to 3d point and line correspondences. *International Journal of Computer Vision*, Vol. 15, pp. 225-243, 1995.
11. Rosenhahn B., Granert O., Sommer G. Monocular pose estimation of kinematic chains. *Applied Geometric Algebras for Computer Science and Engineering, Birkhäuser Verlag*, pp.371–381, 2002.
12. Sommer G., Rosenhahn B., and Zhang Y. Pose estimation using geometric constraints. *In R.Klette, Th. Huang, G.Gimmel'farb (eds.), Multi-Image Search and Analysis*, LNCS 2032, Springer-Verlag, Heidelberg, pp. 153–170, 2001.
13. Sommer G., editor. Geometric Computing with Clifford Algebra. *Springer-Verlag*, Heidelberg, 2001.
14. Walker M.W. and Shao L. Estimating 3-d location parameters using dual number quaternions. *CVGIP: Image Understanding*, Vol. 54:3, pp.358–367, 1991.