O

Object Class Recognition (Categorization)

Svetlana Lazebnik
Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL, USA

Synonyms

Generic object recognition

Related Concepts

▸Generic Object Recognition; ▸Object Detection

Definition

Object class recognition (categorization) is the problem of identifying the class membership (e.g., a car, a dog, a toy) of an object contained in a photograph. Categorization is to be distinguished from the recognition of particular object *instances* (e.g., my Toyota Prius, my dog Fluffy, my teddy bear).

Specific categorization tasks include *image classification* (labeling the image with one of n possible object or scene types), *image annotation* (labeling the image with all objects present), *object detection* or *localization* (finding the location and spatial extent of the object in the image), *image parsing* (segmenting and identifying every object and surface class in the image), and *image understanding* or *interpretation* (describing the content of the image in terms of its scene type, the objects present and their relationships, and the action taking place).

Background

Object class recognition has been one of the chief intellectual endeavors of computer vision since the inception of the field in the 1960s. This problem is extremely difficult due to the large amount of variability in images from the same category, including viewpoint and lighting change, intra-class shape and texture variation, nonrigid deformations and articulations, as well as clutter and occlusion. The key challenge of achieving human-level recognition capability is designing category representations that are robust to all these sources of variability and yet are discriminative enough to distinguish between members of different categories. Another challenge is the sheer number of visual categories that can be readily recognized by humans – 10^4–10^5, according to estimates made by cognitive scientists. Accurate and efficient recognition at this scale requires representations that take into account the complex visual, semantic, and structural relationships between different categories.

Theory

At present, there is no single definitive theory of visual categorization for either humans or computers. Throughout its almost 50-year history, the field of object recognition has seen a succession of proposed theories and representations focusing on different aspects of intra- and inter-category variation

K. Ikeuchi (ed.), *Computer Vision*, DOI 10.1007/978-0-387-31439-6,

and reflecting influences from dominant paradigms in closely related fields, including artificial intelligence, cognitive science, and machine learning. The following is a brief and non-exhaustive overview of some of the most influential representational strategies for tackling the categorization problem.

Early in the history of the field, object representations have emphasized shape. The "geometric era" of recognition has been dominated by 3D shape models composed of geometric primitives such as polyhedra and generalized cylinders [14]. In cognitive science, this approach was supported by the theory of recognition by components [1]. The main advantage of 3D geometric models is that they permit a principled treatment of viewpoint change. They are also expressive – most familiar shapes, at a suitable level of abstraction, can be represented with the help of a few simple geometric part types. Geometric methods have shown some success in recognizing instances of texture-free man-made objects based on edges extracted from relatively uncluttered images. However, they ultimately could not overcome several critical limitations, including the need to manually create 3D object models, the lack of good representations for intra-class shape variability, and the brittleness of model-to-image alignment in the presence of complex appearance changes due to texture, reflectance, lighting, and background clutter.

In the mid-1990s, the geometric recognition paradigm has given way to a radically different philosophy of *appearance-based methods*. The idea of such methods, e.g., *appearance manifolds* [15], is to essentially represent each image by the 1D vector of its pixel values to automatically acquire or *learn* a model of the category by sampling the "manifold" of all possible images from that category under all relevant sources of variability, and to perform recognition by computing the distances between a test image and the learned manifolds. The advantage of this strategy lies in its simplicity and its emphasis on empirical or statistical modeling of pixel-level appearance phenomena that were almost completely ignored by the geometric methods. Its drawbacks include a lack of invariance to any conditions not explicitly sampled at training time (e.g., clutter and occlusion) and a near-lack of abstraction from the image.

The above shortcomings were alleviated with the advent of *local* appearance-based models. The basic building blocks of such models are appearance descriptors computed in the neighborhood of *local features* that are invariant to scaling, rotation, or affine deformations of the image [19]. In addition to having good repeatability and invariance properties, the small spatial extent of local features allows them to survive changes that affect significant parts of the image, such as clutter and occlusion. Even though local invariant features were initially developed for wide-baseline matching and object instance recognition, their usefulness for categorization became apparent very quickly. An important class of local appearance-based models are *bags of features* [2], which work by quantizing feature descriptors to discrete *visual words* and representing images by their visual word histograms. When combined with modern machine learning methods, bags of features can give very good performance on image classification and annotation tasks. However, because they discard all spatial information and do not attempt to associate individual features with objects or background, they are not suitable for localization.

Local features have also been successfully used to construct part-based object models. For example, *constellation models* [20] capture both the appearance and the spatial configuration of object features or parts in a generative Bayesian framework. The probability distributions over part descriptors and locations can be learned in a *weakly supervised* fashion from images where the object of interest is known to be present, but it is not known which features belong to the object. In the formulation of [20], the joint distribution of all part locations is assumed to be Gaussian, which effectively restricts the models to single aspects of rigid objects (e.g., side views of cars). In addition, the computational demands of Bayesian learning and inference in the weakly supervised setting limit the number of parts and the amount of clutter that can be tolerated.

A related important class of part-based object models are *pictorial structures* [8]. Similarly to constellation models, pictorial structures are based on a Bayesian formalism where the probability of an object configuration given an image is a product of individual part likelihoods and a prior over the part locations. Unlike constellation models, pictorial structures constrain the spatial dependencies between parts to form a tree, enabling efficient algorithms for finding the globally optimal match of a pictorial structure

model to an image. The tree structure of these models makes them especially suitable for estimating part layouts for articulated objects such as humans and animals.

When a localization task calls only for an estimate of the object's bounding box, it is common to use a *sliding window detector*. Discriminatively trained detectors using powerful features such as *histograms of oriented gradients* (HOG) [3] work well for certain classes, including faces, cars, and pedestrians. Successful detection of a wider range of highly variable, deformable classes requires a more flexible part-based representation that is yet efficient enough for sliding window search. A promising development in this direction is given by discriminatively trained hierarchical part-based models [9]. These models consist of HOG "filters" at two different levels of resolution for the whole object and its parts, and they support efficient learning and inference in the framework of *latent support vector machines*.

The methods discussed above may be capable of finding part layouts or bounding boxes, but they do not precisely segment out the recognized objects. Historically, it has been hoped that object regions could be accurately segmented by bottom-up *perceptual organization* techniques and then passed along to a recognition module. However, the hope for high-quality bottom-up segmentation has not yet been realized, and today, an emerging consensus in the recognition community holds that segmentation and categorization processes should feed into each other to jointly determine object identity and spatial extent. For example, in the *implicit shape model* approach [12], feature matches "vote" for object hypotheses using a generalized Hough transform, and the winning hypothesis is used to infer a top-down probabilistic segmentation. In turn, the segmentation helps to verify the object hypothesis by discarding the influence of background features.

A different line of attack is required for the problem of image parsing, which involves labeling every pixel in the image by its object class. In this setting, *conditional random fields* provide an appropriate framework for combining local evidence of the presence of multiple object classes with information about their contextual interaction (see, e.g., [17]). In general, the subject of context in recognition is becoming more and more important today as the community moves towards interpreting complex scenes consisting of multiple object types [10].

By embracing statistical appearance-based modeling of intra-class variation, recognition has made great strides over the last two decades. Unfortunately, the original emphasis on representing the 3D nature of real object categories has been largely abandoned in the process. Recently, some researchers have begun an important effort to infuse explicit modeling of viewpoint relationships back into statistical category models [16].

Another topic of much current interest is constructing efficient recognition architectures for large numbers of categories. To date, the standard approach to many-class problems is still to learn separate models for all classes of interest and then to apply each model to the test image. This approach is unsatisfactory because its testing time scales linearly with the number of categories and because its model structure ignores the semantic and visual relationships between different categories. To make problems with many classes more scalable, one potential strategy is to force related classes to share features [18], and another is to organize object models into semantic or visual hierarchies (see, e.g., [13]). Yet another possible way of sharing information between related category models is through attribute-centric representations [6]. An important advantage of such representations is in allowing the recognition system to make useful inferences about previously unseen categories. For example, the system may not have been trained to recognize a tapir in a picture, but it may still be able to infer that it is a four-legged herbivorous mammal.

Application

Automated recognition systems have the potential to make a significant impact in many applications, including search and indexing of large-scale image and video collections, information access through images, human-computer interaction, environmental monitoring, documentation and preservation of natural diversity and cultural heritage, assistance to the visually impaired, personal robotics, automated driving, surveillance and security, and many others.

Open Problems

Despite the rapid progress of the field, designing recognition systems that approach human-level performance on 10^4–10^5 categories is still an open problem. While the biological mechanisms of recognition themselves remain poorly understood, whatever representations the human brain uses give rise to some remarkable abilities, including the ability to infer the properties of never before seen categories and to learn their models from as little as one image, or the ability to construct narratives about the content of an image. Designing computer systems capable of such behavior remains a wide-open goal.

Experimental Results

Until the late 1990s, due to the limitations of digital imaging and computer technology, the evaluation of object recognition methods has tended to be mostly small-scale and anecdotal. In the last few years, the field has established a number of benchmark datasets that enable systematic measurement of progress and comparative evaluation of different algorithms. One of the earliest many-category benchmarks was Caltech-101 [7]. Despite having some widely acknowledged flaws (limited intra-class variation, centered objects, absence of clutter), this dataset had by far the largest *inter-class* diversity at the time of its release, and it has helped to catalyze much research into category-level recognition. More recent datasets include Caltech-256 [11], the PASCAL Visual Object Classes Challenge [5], and ImageNet [4]. In particular, ImageNet contains over 11 million images and over 15,000 concepts, which makes it a good platform for developing and testing new categorization algorithms.

References

1. Biederman I (1987) Recognition by components—a theory of human image understanding. Psychol Rev 94:115–147
2. Csurka G, Dance C, Fan L, Willamowski J, Bray C (2004) Visual categorization with bags of keypoints. In: ECCV international workshop on statistical learning in computer vision, Prague
3. Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. IEEE Conf Comput Vis Pattern Recognit (CVPR) 1:886–893
4. Deng J, Dong W, Socher R, Li LJ, Li K, Fei-Fei L (2009) ImageNet: a Large-Scale Hierarchical Image Database. In: IEEE conference on computer vision and pattern recognition (CVPR), Miami, pp 248–255
5. Everingham M, Van Gool L, Williams CKI, Winn J, Zisserman A (2010). The Pascal visual object classes (VOC) challenge. Int J Comput Vis 88(2):303–338. http://pascallin.ecs.soton.ac.uk/challenges/VOC/
6. Farhadi A, Endres I, Hoiem D, Forsyth D (2009) Describing objects by their attributes. In: IEEE conference on computer vision and pattern recognition (CVPR), Miami, pp 1778–1785
7. Fei-Fei L, Fergus R, Perona P (2004) Learning generative visual models from few training examples: an incremental Bayesian approach tested on 101 object categories. In: IEEE computer vision pattern recognition (CVPR) workshop on generative-model based vision. http://www.vision.caltech.edu/Image_Datasets/Caltech101
8. Felzenszwalb P, Huttenlocher D (2000) Efficient matching of pictorial structures. IEEE Conf Comput Vis Pattern Recognit (CVPR) 2:66–73
9. Felzenszwalb P, Girshick R, McAllester D, Ramanan D (2010) Object detection with discriminatively trained part based models. IEEE Trans Pattern Anal Mach Intell 32(9):1627–1645
10. Galleguillos C, Belongie S (2010) Context based object categorization: a critical survey. Comput Vis Image Underst 114:712–722
11. Griffin G, Holub AD, Perona P (2006) The Caltech-256. Caltech technical report. http://www.vision.caltech.edu/Image_Datasets/Caltech256/
12. Leibe B, Leonardis A, Schiele B (2008) Robust object detection with interleaved categorization and segmentation. Int J Comput Vis 77(1–3):259–289
13. Marszalek M, Schmid C (2008) Constructing category hierarchies for visual recognition. In: European conference on computer vision (ECCV), Marseille, 479–491
14. Mundy J (2007) Object recognition in the geometric era: a retrospective. In: Ponce J, Hebert M, Shmid C, Zisserman A (eds) Toward category-level object recognition. Lecture notes in computer science. Springer, Berlin/New York, pp 3–29
15. Murase H, Nayar S (1995) Visual learning and recognition of 3d objects from appearance. Int J Comput Vis 14(1):5–24
16. Savarese S, Fei-Fei L (2010) Multi-view object categorization and pose estimation. Studies in computational intelligence: computer vision, vol 285/2010. Springer, Berlin/New York, pp 205–231
17. Shotton J, Winn J, Rother C, Criminisi A (2009) TextonBoost for image understanding: multi-class object recognition and segmentation by jointly modeling appearance, shape and context. Int J Comput Vis 81(1):2–23
18. Torralba A, Murphy KP, Freeman WT (2007) Sharing visual features for multiclass and multiview object detection. IEEE Trans Pattern Anal Mach Intell 29(5):854–869
19. Tuytelaars T, Mikolajczyk K (2008) Local invariant feature detectors: a survey. Found Trends Comput Graph Vis 3:177–820
20. Weber M, Welling M, Perona P (2000) Unsupervised learning of models for recognition. In: European conference on computer vision (ECCV), Dublin, pp 18–32

Object Detection

Yali Amit[1] and Pedro Felzenszwalb[2]
[1]Department of Computer Science, University of Chicago, Chicago, IL, USA
[2]School of Engineering, Brown University, Providence, RI, USA

Definition

Object detection involves detecting instances of objects from a particular class in an image.

Background

The goal of object detection is to detect all instances of objects from a known class, such as people, cars, or faces in an image. Typically, only a small number of instances of the object are present in the image, but there is a very large number of possible locations and scales at which they can occur and that need to somehow be explored.

Each detection is reported with some form of *pose* information. This could be as simple as the location of the object, a location and scale, or the extent of the object defined in terms of a bounding box. In other situations, the pose information is more detailed and contains the parameters of a linear or nonlinear transformation. For example, a face detector may compute the locations of the eyes, nose, and mouth, in addition to the bounding box of the face. An example of a bicycle detection that specifies the locations of certain parts is shown in Fig. 1. The pose could also be defined by a three-dimensional transformation specifying the location of the object relative to the camera.

Object detection systems construct a model for an object class from a set of training examples. In the case of a fixed rigid object, only one example may be needed, but more generally multiple training examples are necessary to capture certain aspects of class variability.

Object detection methods fall into two major categories, *generative* [1–5] and *discriminative* [6–10]. The first consists of a probability model for the pose variability of the objects together with an appearance model: a probability model for the image appearance conditional on a given pose, together with a model for background, that is, nonobject images. The model parameters can be estimated from training data and the decisions are based on ratios of posterior probabilities. The second typically builds a classifier that can discriminate between images (or sub-images) containing the object and those not containing the object. The parameters of the classifier are selected to minimize mistakes on the training data, often with a regularization bias to avoid overfitting.

Other distinctions among detection algorithms have to do with the *computational* tools used to scan the entire image or search over possible poses, the type of image *representation* with which the models are constructed, and what type and how much training data is required to build a model.

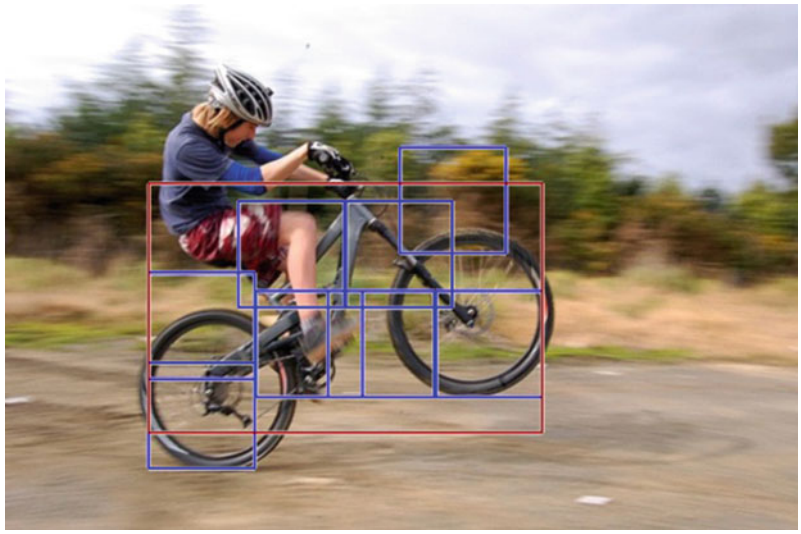
Theory

Images of objects from a particular class are highly variable. One source of variation is the actual imaging process. Changes in illumination and changes in camera position, as well as digitization artifacts, all produce significant variations in image appearance, even in a static scene. The second source of variation is due to the intrinsic appearance variability of objects within a class, even assuming no variation in the imaging process. For example, people have different shapes and wear a variety of clothes, while the handwritten digit seven can be written with or without a line through the middle, with different slants, stroke widths, etc. The challenge is to develop detection algorithms that are *invariant* with respect to these variations and are computationally efficient.

Invariance

The brute force (naive) approach to invariance assumes training data is plentiful and represents the entire range of object variability. Invariance is implicitly learned from the data while training the models.

When training data is limited, it is necessary to build invariance into the models. There are two complementary methods to achieve this. One involves computing invariant functions and features; the other involves searching over latent variables. Most algorithms contain a combination of these approaches. For example, many algorithms choose to apply local transformations



Object Detection, Fig. 1 A bicycle detection specified in terms of the locations of certain parts

to pixel intensities in such a way that the transformed values are invariant to a range of illumination conditions and small geometric variations. These local transformations lead to *features* and the array of feature values is the *feature map*. More significant transformations are often handled through explicit search over latent variables or by learning the remaining variability from training data.

Invariant Functions and Features

This method constructs functions of the data that are invariant with respect to the types of variability described above and can still distinguish between object and background images. This may prove difficult if object variability is extensive. Invariant functions that produce the same output no matter the pose and appearance of the object necessarily have less discriminative power.

There are two common types of operations leading to invariant functions. The first involves computing local features that are invariant to certain image transformations. The second operation involves computing geometric quantities that are invariant to some or all three-dimensional pose variation. For example, the cross ratio among distinguished points is a projective invariant that has been used to recognize rigid objects.

An example of a local feature, invariant to certain photometric variations and changes in illumination, is the *direction* of the image gradient, from which a variety of *edge* features can be computed. More complex features capture the appearance of small image patches and are often computed from edge features. An example would be the histogram of gradient (HOG) features [9]. These features are usually computed at a *dense* grid of locations in the image, leading to a dense feature map.

Local pooling of features is commonly used to introduce some degree of invariance to small geometric variations. A typical example is the MAX operation [1, 11]. In this case, a quantity that is to be computed at a pixel is replaced by the maximum of the quantity in a neighborhood of the pixel. When the maximum is extended over the entire window, the result is a *bag of features* model [12], which reduces to counting the number of binary features of different types that occur within a window. In this case, all spatial information is lost, leading to models that are invariant to fairly large geometric transformations.

For computational reasons, it is often useful to sparsify the feature map by applying local decisions to find a small set of *interest points*. The assumption is that only certain features are useful (or necessary) for object detection. The approach yields *sparse* feature maps that can be processed much more efficiently. Examples of commonly used sparse features are SIFT descriptors [13], corner detectors, and edge conjunctions [1]. One drawback of sparse features is that hard decisions are being made on their presence, and if some are missed, an algorithm may fail to detect an instance of the object.

Note that it is possible to predefine a very large family of features that is never fully computed, rather,

in training an informative subset is selected that can produce the required classification for a particular object class. One example are the Haar features that compute differences of intensity averages in adjacent rectangles of varying sizes and locations [7]. Another example are geometric edge arrangements of increasing complexity.

Latent Variables

An explicit parameterization of the variability can be defined via *latent* variables that are not directly observable from the image data. These are not necessarily needed for the final report on the object detections, but their values simplify the solution of the detection problem. For example, to detect faces at a range of orientations, at each candidate region, one could decide, for *each* possible orientation, whether or not the region contains a face at that orientation. In general a set Θ defines latent parameters that could capture global illumination parameters, a linear or nonlinear map from a model domain into the image domain, or specify the locations of a finite set of object parts. The last case is common in part-based models where latent part placements are used to decide if the object might be present at a particular location in the image [10]. The set of possible latent values, Θ, can be quite large or infinite. This leads to computational challenges that have been addressed by a variety of methods including coarse-to-fine computation, dynamic programming, and geometric alignment.

Detection via Classification

Both generative and discriminative models start with an initial choice of image features and with a choice of the latent pose parameters that will be explicitly modeled. The primary differences between generative and discriminative models are in the methods of training and computation. One important distinction is that generative models do not need data from background to train the object model, whereas discriminative methods need data from both classes to learn the decision boundaries.

The most common approach to object detection reduces the problem to one of binary classification. Consider the problem of detecting objects of fixed size but varying positions in the image. Let W denote a reference window size that an instance of the object would occupy. Let L denote a grid of locations in the image. Let X_{s+W} denote the image features in a window (sub-image) with top-left corner at $s \in L$. One can reduce the detection problem to a binary classification problem as follows. For each location $s \in L$ classify X_{s+W} into two possible classes corresponding to windows that contain an object and windows that do not contain an object. The sliding-window approach to object detection involves explicitly considering and classifying every possible window. Note that the same approach can be used to detect objects of different sizes by considering different window sizes or alternatively windows of fixed size at different levels of resolutions in an image pyramid.

Generative Models

A general framework for object detection using generative models involves modeling two distributions. A distribution $p(\theta; \eta_p)$ is defined on the possible latent pose parameters $\theta \in \Theta$. This distribution captures assumptions on which poses are more or less likely. An appearance model is defined describing the distribution of the image features in a window *conditional* on the pose, $p(X_{s+W}|object, \theta; \eta_a)$. Here, η_p and η_a are the model parameters. For example, η_a might define a *template* specifying the probability of observing certain features at each location in the detection window under a canonical choice for the object pose, while θ specifies a transformation of the template. Warping the template according to θ leads to probabilities for observing certain features at each location in X_{s+W} conditioned on this particular choice of pose parameters [1, 2, 4].

Training data with images of the object are used to estimate the parameters η_p and η_a. Note that the images do not normally come with information about the latent pose variables θ, unless annotation is provided. Estimation thus requires inference methods that handle unobserved variables, for example, the different variants of the expectation maximization algorithm [3, 4]. In some cases, a probability model for background images is estimated as well using large numbers of training examples of images not containing the object.

The basic detection algorithm then scans each candidate window in the image, computes the most likely pose under the object model, and obtains the "posterior odds," that is, the ratio between the conditional probability of the window under the object hypothesis at the optimal pose and the conditional probability of the

window under the background hypothesis. This ratio is then compared to a threshold τ to decide if the window contains an instance of the object

$$\frac{p(X_{s+W}|\text{object}, \theta; \eta_a)\, p(\theta; \eta_p)}{p(X_{s+W}|\text{background})} > \tau.$$

When no background model has been trained offline, a simple *adaptive* background model can be estimated online for each window being tested. In this case, no background training data is needed [4]. Alternative background models involve subcollections of parts of the object model [14].

Discriminative Models

If no explicit latent pose variables are used, the underlying assumption is that the training data is sufficiently rich to provide a sample of the entire variation of object appearance. The discriminative approach trains a standard two-class classifier using large amounts of data from the object and background classes. Many classifier types have been used, including neural networks, SVMs, boosted decision trees, and radial basis functions.

Cascades

Because of the large size of the background population and its complexity, discriminative methods are often organized in *cascades* [7]. An initial classifier is trained to distinguish between the object and a manageable amount of background data. The classifier is designed to have no false negatives at the price of a larger number of false positives. Then a large number of background examples are evaluated and the misclassified ones are collected to form a new background data set. Once a sufficient number of such false positives is accumulated, a new classifier is trained to discriminate between the original object data and the new "harder" background data. Again this classifier is designed to have no false negatives. This process can be continued several times.

At detection time, the classifiers in the cascade are applied sequentially. Once a window is classified as background, the testing terminates with the background label. If the object label is chosen, the next classifier in the cascade is applied. Only windows that are classified as object by all classifiers in the cascade are labelled as object by the cascade. Note that having no false negatives during training provides no guarantee that in testing instances of the objects won't be missed.

Pose Variables

Certain discriminative models can also be implemented with latent pose parameters [10]. Assume a generic classifier defined in terms of a space of classifier functions $f(x; u)$ parameterized by u. Usual training of a discriminative model consists of solving an equation of the form

$$\min_u \sum_{i=1}^{n} D(y_i, f(x_i; u)) + C(u),$$

for some regularization term $C(u)$ which prevents overfitting and a loss function D measuring the distance between the classifier output $f(x_i; u)$ and the ground truth label $y_i = 1$ for object and $y_i = 0$ for background.

The minimization above can be replaced by

$$\min_u \sum_{y_i=1} \min_{\theta \in \Theta} D(1, f(\theta(x_i); u)) + \sum_{y_i=0} \max_{\theta \in \Theta} D(0, f(\theta(x_i); u)) + C(u).$$

Here $\theta(x)$ defines a transformation of the example x. Intuitively for a positive example, one would like there to be some transformation under which x_i is classified as object, while for a negative example, one would like it to be the case that there is no transformation under which x_i is classified as object.

Computational Methods

The basic detection process consists of scanning the image lattice and at each location s testing whether X_{s+W} is classified as object or background. This is typically done at multiple resolutions of the image pyramid to detect objects at multiple scales and is clearly a very intensive computation. There are a number of methods to make it more efficient.

Sparse Features

When sparse features are used, it is possible to focus the computation only in regions around features. The two main approaches that take advantage of this sparsity are *alignment* [15] and the *generalized Hough*

transform. Alignment uses information regarding the relative locations of the features on the object. In this case, the locations of some features determine the possible locations of the other features. Various search methods enable a quick decision on whether a sufficient number of features were found to declare object or not. The Hough transform typically uses information on the location of each feature type relative to some reference point in the object. Each detected feature votes with some weight for a set of candidate locations of the reference point. Locations with a sufficiently large sum of weighted votes determine detections. This idea can also be generalized to include identification of scale as well. The voting weights can be obtained either through discriminative training or through generative training [1].

Cascades

As mentioned above, the cascade method trains a sequence of classifiers with successively more difficult background data. Each such classifier is designed to be very computationally efficient. When the data in the window X_{s+W} is declared background by any classifier of the cascade, the decision is final and the computation proceeds to the next window. Since most background windows are rejected early in the cascade, most of the windows in the image are processed very quickly.

Coarse to Fine

The cascade method can be viewed as a coarse-to-fine decomposition of background that gradually makes finer and finer discriminations between object and background images that have significant resemblance to the object. An alternative is to create a coarse-to-fine decomposition of object poses [8]. In this case, it is possible to train classifiers that can rule out a large subset of the pose space in a single step. A general setting involves a rooted tree where the leaves correspond to individual detections and internal nodes store classifiers that quickly rule out all detections below a particular node. The idea is closely related to branch-and-bound methods [12] that use admissible lower bounds to search a space of transformations or hypotheses.

Dynamic Programming

There are a number of object detection algorithms that represent objects by a collection of parts arranged in deformable configurations or as hierarchies of such arrangements of parts of increasing complexity. When the hierarchies and the arrangements have the appropriate structure, dynamic programming methods can be used to efficiently search over the spaces of arrangements [1, 2].

Object Detection, Fig. 2
The output of a face detection algorithm

Application

Object detection methods have a wide range of applications in a variety of areas including robotics, medical image analysis, surveillance, and human-computer interaction. Current methods work reasonably well in constrained domains but are quite sensitive to clutter and occlusion.

A popular benchmark for object detection is the PASCAL VOC object detection challenge. The goal of the challenge is to detect objects from common categories such as people, cars, horses, and tables in photographs. The challenge has attracted significant attention in the computer vision community over the last few years, and the performance of the best systems has been steadily increasing by a significant amount on a yearly basis.

Face detection is a typical application of object detection algorithms. There has been significant success in deploying face detection methods in practical situations. For example, current digital cameras use face detection to decide where to focus and even detect smiles to decide when to take the picture. Figure 2 shows a typical output of a face detection algorithm.

References

1. Amit Y (2002) 2d object detection and recognition: models, algorithms and networks. MIT Press, Cambridge
2. Felzenszwalb P, Huttenlocher D (2005) Pictorial structures for object recognition. Int J Comput Vis 61(1):55–79
3. Fergus R, Perona P, Zisserman A (2003) Object class recognition by unsupervised scale-invariant learning. In: IEEE conference on computer vision pattern recognition (CVPR 2003), Madison
4. Amit Y, Trouvé A (2007) POP: patchwork of parts models for object recognition. Int J Comput Vis 75(2):267–282
5. Jin Y, Geman S (2006) Context and hierarchy in a probabilistic image model. In: IEEE conference on computer vision pattern recognition (CVPR 2006), New York
6. Rowley HA, Baluja S, Kanade T (1998) Neural network-based face detection. IEEE Trans Pattern Anal Mach Intell 20(1):23–38
7. Viola P, Jones MJ (2004) Robust real time face detection. Int J Comput Vis 57(2):137–154
8. Fleuret F, Geman D (2001) Coarse-to-fine face detection. Int J Comput Vis 41(1–2):85–107
9. Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. In: IEEE conference on computer vision pattern recognition (CVPR 2005), San Diego
10. Felzenszwalb P, Girshick R, McAllester D, Ramanan D (2010) Object detection with discriminatively trained part based models. IEEE Trans Pattern Anal Mach Intell 32(9):1627–1645
11. Riesenhuber M, Poggio T (2000) Models of object recognition. Nat Neurosci 3:1199–1204. Supplement
12. Lampert C, Blaschko M, Hofmann T (2009) Efficient subwindow search: a branch and bound framework for object localization. IEEE Trans Pattern Anal Mach Intell 31(12):2129–2142
13. Lowe D (2004) Distinctive image features from scale-invariant keypoints. Int J Comput Vis 60(2):91–110
14. Chang LB, Jin Y, Zhang W, Borenstein E, Geman S (2011) Context computation, and optimal roc performance in hierarchical models. Int J Comput Vis 93:117–140
15. Ullman S (1996) High-level vision. MIT Press, Cambridge

Object Extraction

▸Interactive Segmentation

Object Models

▸Model-Based Object Recognition

Object Motion Blur

▸Motion Blur

Object Parameterizations

▸Model-Based Object Recognition

Object Recognition

▸Line Drawing Labeling

Object Representations

▸Model-Based Object Recognition

Object Segmentation

▸Semantic Image Segmentation

Obstacle Detection

Larry Matthies
Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, USA

Synonyms

Hazard detection

Definition

Obstacle detection is the process of using sensors, data structures, and algorithms to detect objects or terrain types that impede motion.

Background

Obstacle detection is applicable to anything that moves, including robot manipulators and manned or unmanned vehicles for land, sea, air, and space; for brevity, these are all called *vehicles* here. Obstacle detection and hazard detection are synonymous terms, but are sometimes applied in different domains; for example, obstacle detection is usually applied to ground vehicle navigation, whereas hazard detection is often applied to aircraft or spacecraft in the process of landing, as in "landing hazard detection." Obstacle detection is a system problem that encompasses sensors that perceive the world, world models that represent the sensor data in a convenient form, mathematical models of the interaction between objects and the vehicle, and algorithms that process all of this to infer obstacle locations. Obstacle detection algorithms use the world model and the interaction model to locate obstacles, to characterize the degree to which the obstacles impede motion, and to produce an *obstacle map* for use by path planning algorithms. The complexity of obstacle detection systems varies greatly, depending on the domain of operation, the size and cost of the vehicle, and the degree of reliability required.

Because sensor data is noisy and incomplete, often it is necessary to combine many sensor measurements into the world model to reduce noise and fill in gaps in the world model before applying obstacle detection algorithms; this makes obstacle detection related to mapping. Obstacles may completely block motion of the vehicle or just make it more difficult to move, such as having to move slowly over rough ground; this makes obstacle detection related to terrain classification.

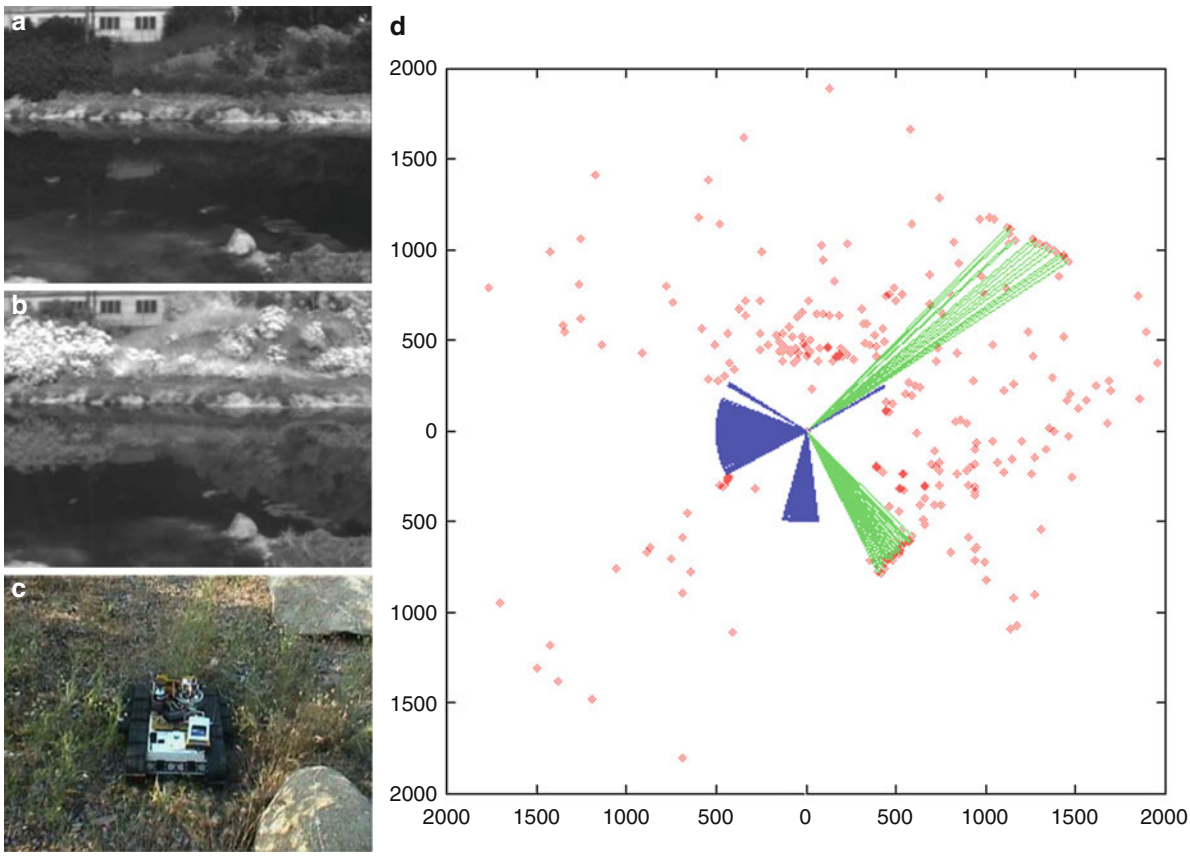
Sensors for Obstacle Detection

An excellent introduction to a broad range of sensors for obstacle detection is included in [1]. 3-D perception plays a central role in obstacle detection, because the geometry of the world is a major factor in determining what constitutes an obstacle. 3-D sensors include active ranging techniques, like ladar, radar, and sonar, and passive ranging techniques, like stereo vision and structure from motion. Appearance can also be useful for discriminating obstacles from non-obstacles, so color, texture, polarization, and radiant temperature can also be important sensor attributes. Since all 3-D sensors have limited range, appearance attributes are especially important for detecting obstacles at long range.

Often, mechanical properties like the stiffness of objects or terrain are important to obstacle detection, as well as geometric properties. This requires sensors that directly or indirectly give insight into mechanical properties of the scene. These can be proprioceptive sensors, like inertial sensors, wheel encoders, or instrumented bumpers, that sense aspects of the vehicle-world interaction as it occurs, or remote sensors that perceive the terrain ahead of the vehicle. Examples of remote sensing to predict terrain mechanical properties are using ladar [2] or multispectral imagery [3] to infer whether material is vegetation that can be pushed through by the vehicle or is something more solid that must be avoided (Fig. 1).

Environmental conditions like ambient illumination and atmospheric obscuration also impact obstacle detection sensors. For example, fog, smoke, and night operation may require active sensors or thermal infrared cameras (Fig. 2), while thick dust or heavy precipitation may require radar. Dynamic environments require sensors that measure the velocity of obstacles as well as their size and location.

Choosing appropriate sensors is a critical first step in obstacle detection, to ensure that the sensors can



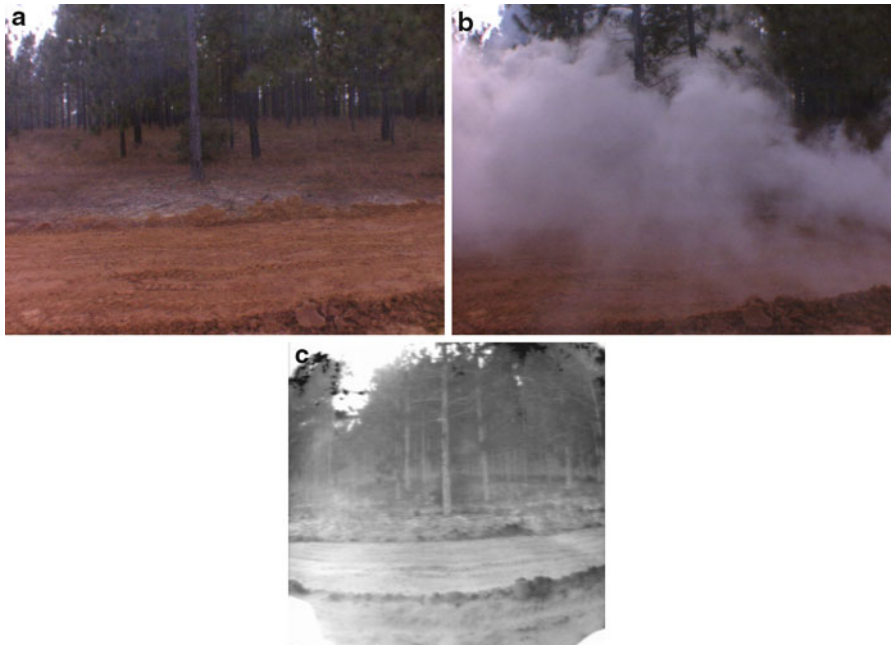
Obstacle Detection, Fig. 1 Sensors for perceiving vegetation. (**a**) Image acquired with narrow spectral band around 650 nm, with a pond in the foreground and soil and vegetation in the background. The foliage is *darker* than the soil in this spectral band. (**b**) Image of the same scene acquired with a narrow spectral band around 800 nm. The foliage is *brighter* than the soil in this spectral band. Ratios of spectral bands at 650 and 800 nm can be used to distinguish growing foliage more effectively than can be done with just visible spectrum imagery. (**c**) A small robot in sparse grass, with a few rocks nearby, carrying a single-axis, 360° scanning laser range finder (ladar). (**d**) Plot of ladar pixels (*red dots*), with sensor at (0,0). *Blue cones* are areas that were blocked by structure on the robot. Pixels that appear to be randomly distributed are measurements of grass; pixels that fall into line segments are measurements of the rocks. The *green rays* are rock pixels that were automatically detected

discriminate obstacles under the required conditions. This can be difficult; for example, operation in high levels of airborne dust requires use of sensors that measure range to solid objects behind the dust, instead of to the dust. It can also be very challenging to obtain appropriate sensors that fit within the size, weight, power, and cost constraints of the application.

Data Structures for Obstacle Detection

Obstacle detection algorithms can be developed to process raw sensor data or data that has been transformed into another representation. Raw data is often expressed in *sensor space* (also called *image space* for some sensors); these are data structures that are indexed by the natural coordinate system of the sensor, such as (row, column) or (azimuth, elevation). With two-axis ladar, for example, surface normals, range discontinuities, and height above ground can be computed directly from range images provided by the sensor, and obstacle locations can be inferred from these quantities. Operating in image space preserves pixel adjacency information, which can be convenient for segmentation.

The normal alternative to working in image space is to work in a *map space*, such as a Cartesian reference frame that is either centered on the vehicle or has a fixed origin in the world. Map representations can simply accumulate raw data from many sensors or



Obstacle Detection, Fig. 2 Ability to see through smoke with thermal infrared imagery. (**a**) Scene viewed with a color camera. (**b**) Same scene with smoke. (**c**) Same scene viewed with the smoke, using a mid-wave infrared camera, which is sensitive to light from 3 to 5 μm; the sensor sees quite well through the smoke

many points in time, such as point clouds from range-imaging sensors like ladar or stereo vision, or they may be gridded or geometric primitives [4]. Elevation maps and occupancy grids are common gridded representations. Elevation maps record the height of the ground surface in each cell of a discretized map of the ground plane. Occupancy grids record a probability that each cell of the map is occupied by solid material, instead of empty; these can be defined in 2-D or 3-D [5, 6]. Geometric representations fit geometric primitives to the raw sensor data, like line segments or polygons for 2-D maps and planar surfaces or bounding boxes for 3-D maps.

Map space representations are used for several purposes. First, they are used to accumulate data over time, which is valuable to "fill in" gaps that may occur in individual frames of sensor data. Second, they facilitate noise reduction, because the accumulated data can be averaged or filtered for outliers. Third, they can effectively interpolate higher-resolution representations of the world than are available from individual frames of sensor data; an example is when occupancy grids are used to build world models from wide-angle sonar data. Finally, they can be in a coordinate system that makes some aspects of obstacle detection easier; for example, transforming range data from image space to a Cartesian map space data structure makes it possible to use shift-invariant algorithms to detect fixed-size obstacles.

Uncertainty is inherent in sensors, therefore also in maps and in obstacle detections, so modeling that uncertainty is an important issue. One of the most common approaches to this is the use of occupancy grids to estimate the probability that map cells contain obstacles [5, 6]. Uncertainty also can drive the choice of world model representation, such as the use of polar-perspective maps, which use an inverse-range parameterization of the map to match the spatial and range resolution characteristics of stereo vision [7].

The output of the obstacle detection process is also represented in some type of world model. Some obstacle models identify only certain locations as obstacles; these can be a subset of the cells of a grid map, polygons in the ground plane, bounding boxes in 3-D, or other representations. For off-road navigation, it is common for obstacle detection to produce a *cost map*, where every cell of the map gets an associated traversal cost [8]. Especially for off-road navigation, obstacle

detection overlaps the problem of trafficability analysis, where the vehicle can pass over a given patch of terrain, but may sink, slip, or experience a rough ride in the process; in this case, the output may represent aspects of trafficability that are useful for path planning [9].

Vehicle-World Interaction Models for Obstacle Detection

Obstacle detection algorithms implicitly or explicitly use a model of how the vehicle interacts with the world to determine where obstacles exist. In the simplest case, the model may be just a binary decision based on raw sensor data, such as when single-axis scanning ladars are used for obstacle detection indoors and any object that produces a range measurement counts as an obstacle. For outdoor navigation, the size of an object affects whether or not it is considered an obstacle; for example, small bumps on the ground may be ignored, but large rocks are obstacles. This distinction depends on characteristics of the vehicle, such as wheel diameter, ground clearance, velocity, and suspension stiffness. In this context, interaction modeling can take into account just the geometry of the vehicle and world, or varying levels of fidelity in the dynamics of the interaction [10, 11].

Algorithms for Obstacle Detection

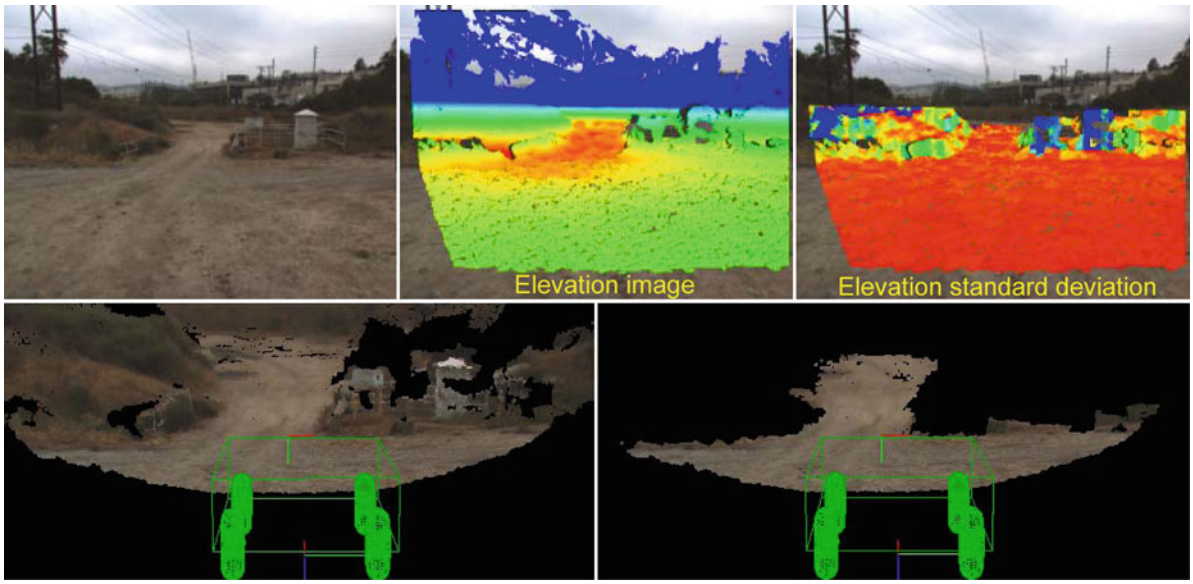
Obstacle detection algorithms use the vehicle-world interaction model to transform the sensor data and the world model into an intermediate representation that enables more efficient path planning. There are many approaches to this, which depend on the application context.

As an example, in a purely geometric approach for ground vehicles using grid-based world models like elevation maps or occupancy grids, local operators are convolved over the map to determine if there is an obstacle at each point in the map. These operators vary from being as simple as thresholding the cell probabilities in occupancy grids, to fitting planes or computing elevation variance over local patches of elevation maps, to configuration tests that tell if a geometric model of the vehicle will fit in a given configuration at a given point in the map without intersecting an object in the world (Fig. 3) [10, 13]. The output of such algorithms is typically a cost map, with a very high cost assigned to cells the vehicle cannot enter and intermediate costs assigned to cells that can be traversed with some degree of difficulty. Similar concepts can be applied to range data in image space. Path planning algorithms then search for low-cost paths through the cost map.

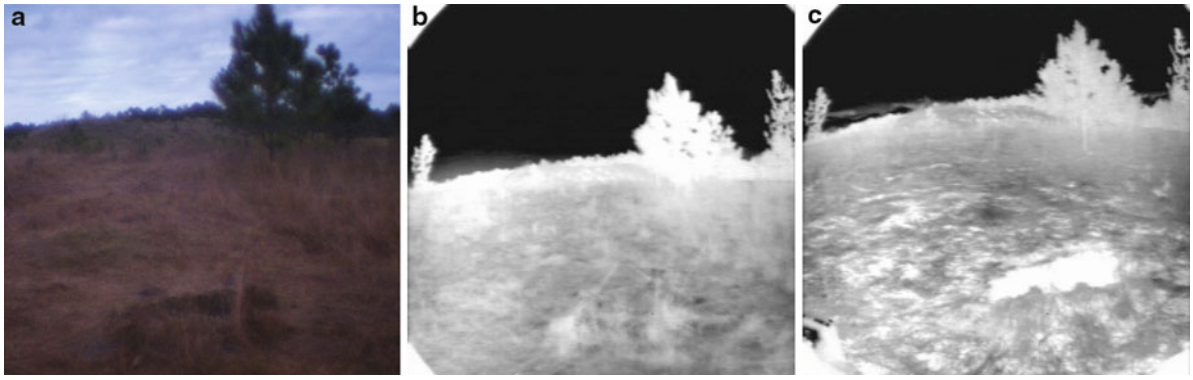
In some contexts, geometry is not the only consideration: material or mechanical properties of objects also determine whether they are obstacles. Common examples are soft objects like vegetation, which vehicles can readily traverse but which may appear as obstacles to naïve interpretations of range data. In this example, heuristics can be designed based on the spatial scatter of 3-D point clouds to estimate whether a patch of range data represents a solid obstacle surface, or a cloud of thin objects most likely to be traversable vegetation, or thin objects more likely to be wires or fences that are obstacles [3, 12]. Using both geometric and appearance cues is important in this context, since color, texture, and other appearance properties of objects may be as important as their size and shape in determining whether or not they are obstacles.

Some classes of obstacles have been detected based on more detailed, physics-based analysis of multiple sensor phenomenologies. *Negative obstacles* are any depression below the ground plane, such as potholes and ditches. These are harder to detect than *positive obstacles*, which protrude above the ground plane, because as their distance increases, negative obstacles grow smaller in the image at a faster rate than positive obstacles. At night, heat transfer phenomena tend to keep the interiors of small negative obstacles warmer than surrounding terrain; therefore, features computed from range data and thermal infrared images can help to detect them (Fig. 4) [14]. Water bodies and patches of ice or snow are another example, where distinctive effects of reflections, absorption, polarization, and temperature can create discriminable features in both range data and specific spectral channels of appearance data [15]. Different materials can have different signatures in radar, so fusion of radar with other sensors also helps to distinguish obstacles from traversable materials.

Early algorithms for obstacle detection often used manually designed procedures based on the designer's knowledge of the problem. For several reasons, there is now a trend to incorporate more automated,



Obstacle Detection, Fig. 3 Some steps in obstacle detection. *Top row*, *left* to *right*: one image of a stereo image pair of a scene; false color overlay of elevation data on the image, with *red showing lowest* elevation and *blue showing highest*; false color overlay of elevation standard deviation on image, as computed from an associated grid map. *Red* is *low* standard deviation, *blue* is *high*. *Bottom row*, *left* to *right*: 3-D visualization of entire point cloud from stereo matching; 3-D visualization of just the ground surface as segmented from the elevation standard deviation (From [13])



Obstacle Detection, Fig. 4 Thermal phenomenology for negative obstacle detection. (**a**) Color image of a pothole at 5 p.m.; the pothole is the *dark* patch in the *bottom center* of the image. (**b**) Thermal infrared image of the same scene at the same time. (**c**) Thermal infrared image of the same scene at 7 a.m.; now differential cooling of the interior of the pothole and the terrain has made the pothole noticeably warmer in the thermal image

learning-based methods. First, terrain classification algorithms have always needed training data; to reduce the human labor involved, it is desirable to automate the process of labeling the training data. Second, the great variety of objects in the world makes it impractical to manually program rules for distinguishing all obstacles. Sometimes both of these issues can be addressed by using the vehicle's own experience, by using knowledge gained from one sensor to automatically generate training labels for data from another sensor. Third, obstacles may be easier to detect at short range than at long range, because richer or higher-resolution perception is possible at short range. All of these issues are embodied in *self-supervised, near-to-far* methods for training obstacle detectors. In this paradigm, the most detailed knowledge of obstacles

comes from proprioceptive sensing of physical contact with them (*underfoot*); this knowledge can be used to train classifiers applied to *near field* remote sensors, where terrain is within range of 3-D perception sensors. In turn, classifiers trained with the aid of 3-D perception in the near field can be used to discriminate obstacle using only appearance features, like color and texture, for terrain beyond the reach of 3-D perception (*far field*) [16].

Kinematic issues enter the obstacle detection problem when assessing the likelihood that the vehicle will collide with other moving objects. This raises the problems of segmentation, motion estimation, and behavior prediction for other objects, which are large topics in their own right. Dynamic issues arise when considering whether the vehicle may slip, sink, roll over, or experience damaging reaction forces while traveling. Assessing these factors requires appropriate vehicle-world interaction models. When these models are expensive to evaluate, instead of doing obstacle detection first over the entire world model, then searching for low-cost paths, obstacle detection may be embedded within the path planner. In this case, the path planner searches for candidate paths that obey kinematic and dynamic constraints on vehicle motion and evaluates obstacle-checking criteria within the process of evaluating the quality of each path [17]. This can be made more efficient by applying a hierarchical, triage process, where simple, low-cost obstacle detection algorithms are used first to find areas that are easily traversable, definitely not traversable, or in between, then using higher-fidelity models as needed in the gray areas. Obstacle detection is often done this way when the planning problem involves many degrees of freedom, such as with robot manipulators.

Open Problems

Some classes of obstacles are still difficult to recognize reliably, such as mud [18] and small negative obstacles. Sensor fusion and modeling context are potential approaches to these problems. Self-supervised, near-to-far training of learning-based obstacle detection methods are relatively new techniques and are under ongoing refinement. Dynamical aspects of vehicle-world interaction modeling are still being actively explored, including the interplay between such models and efficient algorithms for obstacle detection, trafficability evaluation, path planning, and learning. Designing efficient data structures and multi-resolution world models for obstacle detection at both short and long range and over long distance travel remains an important problem. Developing sensor hardware with low size, weight, power, and cost that enables obstacle detection at long range, high speed, and in all environmental conditions will be a long-term challenge.

References

1. Everett HR (1995) Sensors for mobile robots: theory and application. A.K. Peters, Wellesley, MA
2. Matthies L, Bergh C, Castano A, Macedo J, Manduchi R (2005) Obstacle detection in foliage with ladar and radar. In: Dario P, Chatila R (eds) Robotics research: the eleventh international symposium. Springer, Berlin, pp 291–302
3. Matthies L, Kelly A, Litwin T, Tharp G (1996) Obstacle detection for unmanned ground vehicles: a progress report. In: Giralt G (ed) Robotics research: the seventh international symposium. Springer, Berlin, pp 475–486
4. Lalonde J-F, Vandapel N, Hebert M (2007) Data structures for efficient dynamic processing in 3-D. Int J Robot Res 26(8):777–796
5. Moravec H, Elfes AE (1985) High resolution maps from wide angle sonar. In: Proceedings of the IEEE international conference on robotics and automation (ICRA), St. Louis, pp 116–121
6. Wurm KM, Hornung A, Bennewitz M, Stachniss C, Burgard W (2010) OctoMap: a probabilistic, flexible, and compact 3D map representation for robotic systems. In: Proceedings of the workshop on best practice in 3D perception and modeling for mobile manipulation, Anchorage
7. Bajracharya M, Moghaddam B, Howard A, Brennan S, Matthies L (2009) A fast stereo-based system for detecting and tracking pedestrians from a moving vehicle. Int J Robot Res 29(11–12):1466–1485
8. Stentz A, Hebert M (1995) A complete navigation system for goal acquisition in unknown environments. Auton Robot 2(2):127–145
9. Iagnemma K et al (2011) Terramechanics modelling of Mars surface exploration rovers for simulation and parameter estimation. In: Proceedings of the ASME international design engineering technical conference, Washington, DC
10. Olin KE, Tseng DT (1991) Autonomous cross-country navigation: an integrated perception and planning system. IEEE Expert 6(4):16–30
11. Trease B et al (2011) Dynamic modelling and soil mechanics for path planning of the Mars Exploration Rovers. In: Proceedings of the ASME international design engineering technical conference, Washington, DC
12. Lalonde J-F, Vandapel N, Huber DF, Hebert M (2006) Natural terrain classification using three-dimensional ladar data for ground robot mobility. J Field Robot 23(10):839–861

13. Rankin AL, Huertas A, Matthies L (2009) Stereo vision based terrain mapping for off-road autonomous navigation. In: Proceedings of the SPIE symposium on unmanned systems technology XI. Orlando, Florida, USA
14. Matthies L, Rankin A (2003) Negative obstacle detection by thermal signature. In: Proceedings of the IEEE/RSJ conference on intelligent robots and systems (IROS), Las Vegas
15. Matthies L, Bellutta P, McHenry M (2003) Detecting water hazards for autonomous off-road navigation. In: Proceedings of the SPIE symposium on unmanned ground vehicles V. Orlando, Florida, USA
16. Bajracharya M, Howard A, Matthies L, Tang B, Turmon M (2009) Autonomous off-road navigation with end-to-end learning for the LAGR program. J Field Robot 26(1):3–25
17. Howard TM (2009) Adaptive model-predictive motion planning for navigation in complex environments. Ph.D. thesis, Carnegie Mellon University, CMU-RI-TR-09-32
18. Rankin A, Matthies L (2010) Passive sensor evaluation for unmanned ground vehicle mud detection. J Field Robot 27(4):473–490

Occlusion Boundaries from Motion

▸Occlusion Detection

Occlusion Boundaries from Stereo Matching

▸Occlusion Detection

Occlusion Detection

Rodrigo Benenson
K.U. Leuven departement Elektrotechniek – ESAT, Centrum voor beeld- en spraakverwerking – PSI/VISICS, Heverlee, Belgium

Synonyms

Occlusion boundaries from motion; Occlusion boundaries from stereo matching; Foreground-background assignment

Related Concepts

▸Edge Detection; ▸Object Detection; ▸Occlusion Handling

Definition

Occlusion detection refers to the set of techniques employed to detect which areas of the images are occlusion boundaries or areas that appear occluded in views of the scene.

Background

Occlusion is one of the fundamental phenomenon that limits the information available in an image. Given a 3-dimensional scene containing non transparent objects, a projection of the scene into the image will not include information about all the surfaces in the scene. The backside of the nontransparent objects will be occluded by the frontside, and the foreground objects will occlude the background surfaces. Which areas are occluded and which ones are visible depends on the position of the camera with respect to the scene.

In natural images, occlusions are prevalent. Knowing which are the occlusion boundaries in an image helps segmenting pixels at the object's level and thus help processes such as objects detection or relative depth estimation. The occlusion areas depend on the camera position; thus, most algorithms matching two or more images are concerned with occlusion detection; since not all the surface points visible in image I_1 will be visible in image I_2, not all pixels in I_1 will have a corresponding pixel I_2. Due to this stereo matching, optical flow and object tracking algorithms are particularly concerned with occlusion detection.

Theory

For occlusion detection two cases should be distinguished: single image or multiple images. When a single image is available, no direct observation of occlusion boundaries can be done, and thus, the nature of the problem is quite different from the multiple images case.

Single Image

If the depth assigned to each pixel was known, then the occlusion boundaries would correspond to where the depth is discontinuous. Without any additional information but a single image, estimating the

occlusion boundaries corresponds to estimating depth discontinuities when depth information is not available. Without priors, such problem cannot be solved. A common assumption (in stereo matching and optical flow estimation literature) is that depth discontinuities generate color discontinuities. Classical edge detection algorithms can be used to find the color discontinuities and the occlusion detection problem becomes a classification problem; given the detected edges of the image, one wants to know which one corresponds to depth discontinuities and which ones correspond to texture gradient over the same surface. Such classification problem can be addressed using machine learning techniques such as the one described in [1].

Multiple Images

When multiple images are available, there are means to access partial depth information. Such information allows to have a more reliable estimate of the occlusion boundaries. Multiple images of the same scene may come from multiple cameras observing the scene (multi-view imaging, stereo imaging) or from one moving camera observing the scene at two consecutive instants.

Detecting Occlusion Areas via Pixel Matching

When multiple views of the scene are available, it is possible to do pixel-level matching between the multiple images. This is an instance of a data association problem, and it is commonly instantiated as a labeling problem in the vision community. Each pixel in image I_1 will be either visible in I_2, and thus should be matched or it may be part of an occluded area in I_2 and thus will be rejected during the matching. This strategy is effective for stereo matching (multi-view can be reduced to a set of pair-wise matching problems), and for optical flow cases.

Simultaneously deciding which is the displacement/disparity of each pixel and which pixel is occluded or not is a difficult optimization problem (see, for instance, [2]). A common strategy to relax the problem is to focus on symmetry constraints. Non-occluded pixels are visible in both images; by definition occluded pixels are visible in only one image or none. Thus, a common strategy to detect occluded pixels consists on matching first I_1 to I_2 (disregarding occlusion issues), then matching I_2 to I_1, and finally verifying which displacements are consistent between the two matching results. Occluded pixels are expected to have inconsistent displacement.

This strategy has shown acceptable results for stereo matching and optical flow problems [3, 4].

Detecting Occlusion Boundaries via Flow Cues

In the case of optical flow, there are additional cues available other than simply pixel matching. Due to parallax effects or actual object motion, different objects will appear in the image with different (2-dimensional) motion vectors. Assuming object rigidity, it is then possible to delineate objects boundaries by finding areas where the optical flow is divergent [5]. Such objects boundaries estimates are also occlusion boundaries estimates.

Detecting Occlusions During Object Tracking

Previous sections describe occlusion detection at the pixel level. Occlusion detection is also a common concern when trying to track whole objects across multiple frames (i.e., trying to solve the object tracking problem). In its common formulation objects tracking is a data association problem of the same kind as pixel-level matching for depth estimation or optical flow. In its simplest form object-level occlusion detection will be done either via a threshold on the appearance similarity (if the appearance of the expected object location is too different from the object template, the object is considered occluded) or via a temporal threshold on the objects detector (if an object nearby the expected location has not been detected for too long, it is considered occluded) [6].

Application

When computing depth maps, optical flow, or objects tracks for decision making, it is important to know which areas are reliable and which ones are not. By definition occluded areas offer unreliable information (no data) and thus must be detected. Once unreliable areas are detected, the decision process is improved. In applications when the output must be complete (e.g., in computer graphics), knowing which areas are occluded, enables applying infilling/interpolation algorithms over them to provide pleasing results. Occlusion areas and occlusion boundaries are linked to

object boundaries, and as such are also an useful clue for object segmentation and detection.

References

1. Stein A (2008) Occlusion Boundaries: Low-Level Detection to High-Level Reasoning. PhD thesis, Robotics Institute, Carnegie Mellon University
2. Strecha C, Fransens R, Van Gool L (2004) A probabilistic approach to large displacement optical flow and occlusion detection. Stat Methods Video Process 3247:25–45
3. Sun J, Li Y, Kang S, Shum HY (2005) Symmetric stereo matching for occlusion handling. In: Proceedings of the IEEE Conference Computer Vision and Pattern Recognition (CVPR)
4. Alvarez L, Deriche R Dr, Papadopoulo T, Sánchez J (2007) Symmetrical dense optical flow estimation with occlusions detection. Int J Comput Vis 75(3):371–385
5. Thompson WB, Mutch KM, Berzins VA (1985) Dynamic occlusion analysis in optical flow fields. IEEE Trans Pattern Anal Mach Intell 7(4):374–383
6. Yilmaz A, Javed O, Shah M (2006) Object tracking: a survey. Comput Surv 38(4):13.1–13.45

Occlusion Handling

Rodrigo Benenson
K.U. Leuven departement Elektrotechniek – ESAT, Centrum voor beeld- en spraakverwerking – PSI/VISICS, Heverlee, Belgium

Related Concepts

►Inpainting; ►Occlusion Detection

Definition

Occlusion handling refers to the set of techniques employed to mitigate the effects of occlusion when doing inference from image.

Background

Occlusion is a fundamental phenomenon that limits the information that can be extracted from a camera observing a scene. Either when computing depth maps, optical flow, or tracking objects, occlusion will interfere with the inference process and create blank areas. In many application, simply knowing which areas are occluded is good enough for the desired output. In other applications, it is desired to have an infilled/interpolated output without any blank. Occlusion handling makes reference to the latter case.

Theory

How to tackle the occluded areas is an application-specific problem. The three most common tasks that have to deal explicitly with occlusion are stereo matching, optical flow, and objects tracking. Other applications, such as visual odometry, also suffer from occlusion, but they simply ignore it and consider it as input noise.

Occlusion Handling in Pixel Matching

For stereo matching and optical flow, the common strategy is to do extrapolation. Due to occlusion only a partial (non-occluded) view of the background object/surface is available. It is commonly assumed that the background surface is large, and thus covers also the occluded area. Given this assumption, the occluded area is simply infilled using extrapolations from the surrounding connecting background area.

For stereo matching, this can be done by simply infilling row per row the occluded area using the same disparity value as the lowest one (farthest from the camera) between the right and left side of the occlusion area. More sophisticated approaches try to fit local planes to infill the disparity map [1, Chap. 5].

For optical flow, the inpainting is can be done via an edge-sensitive anisotropic smoothing, that will naturally infill the blank areas using the surrounding flow [2].

Occlusion Handling in Object Tracking

In object tracking, one must distinguish the online case (only previous frames are available) and the offline case (all image frames of a video sequence are available).

For the online case, the appearance similarity is the main clue available. When no data association is found between a current track and the elements in the current frame, then the object position is extrapolated using the motion model. If too much time passes since the last successful detection, then the track is aborted. Should

the object reappear later in the video, it will be annotated with a different object identifier (identity switch) unless a life-long set of detected objects is kept, and the match between the re-apparition and the object model is strong [3]. When objects are re-detected and associated after occlusions, the track trajectory can be adjusted based on the new evidence [4].

For the offline case, a global optimization problem can be cast [5]. In such setup, the time of occlusion is trade-off with the strength of the global appearance similarity; it is then possible to track objects through longer occlusions without loosing the object identifier.

References

1. Bleyer M (2006) Segmentation-based Stereo and Motion with Occlusions. PhD thesis, Vienna University of Technology
2. Ince S, Konrad J (2008) Occlusion-aware optical flow estimation. IEEE Trans Image Process 17:1443–1451
3. Yilmaz A, Javed O, Shah M (2006) Object tracking: a survey. Comput Surv 38(4):13.1–13.45
4. Leibe B, Schindler K, Cornelis N, Van Gool L (2008) Coupled detection and tracking from static cameras and moving vehicles. PAMI 30(10):1683–1698
5. Xing J, Ai H, Lao S (2009) Multi-object tracking through occlusions by local tracklets filtering and global tracklets association with detection responses. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)

Omnidirectional Camera

Davide Scaramuzza
Artificial Intelligence Lab - Robotics and Perception Group, Department of Informatics, University of Zurich, Zurich, Switzerland

Synonyms

Catadioptric camera; Fisheye camera; Panoramic camera; Spherical camera; Wide-angle camera

Related Concepts

►Camera Calibration; ►Camera Parameters (Intrinsic, Extrinsic); ►Epipolar Geometry; ►Intrinsic Parameters; ►Omnidirectional Vision; ►Structure-from-Motion (SfM)

Definition

An omnidirectional camera (from *omni*, meaning all) is a camera with a 360-degree field of view in the horizontal plane or with a visual field that covers a hemisphere or (approximately) the entire sphere.

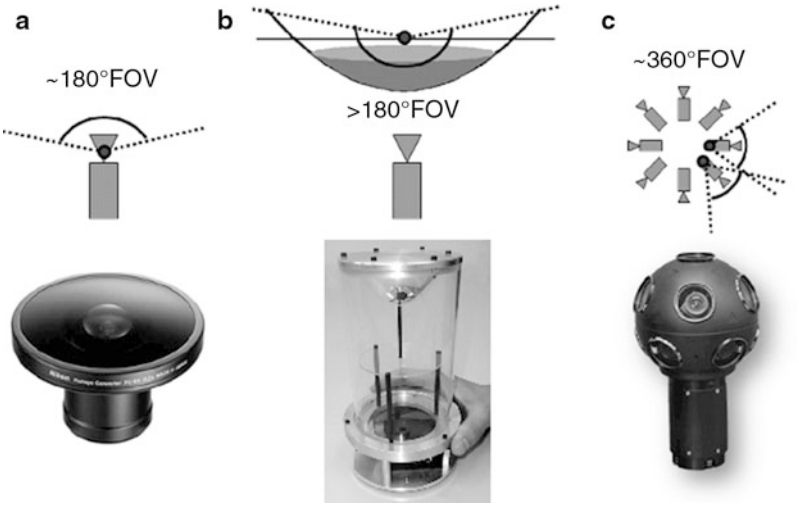
Background

Most commercial cameras can be described as pinhole cameras, which are modeled by a perspective projection. However, there are projection systems whose geometry cannot be described using the conventional pinhole model because of the very high distortion introduced by the imaging device. Some of these systems are omnidirectional cameras.

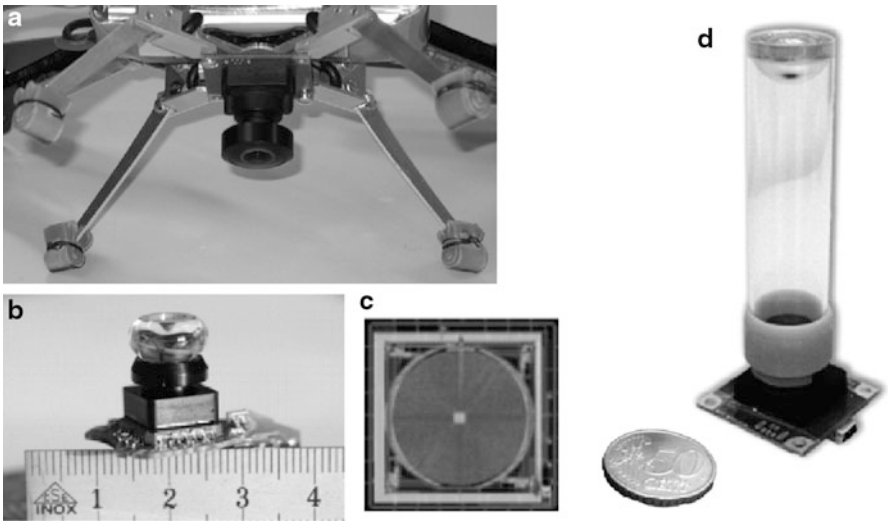
There are several ways to build an omnidirectional camera. Dioptric cameras use a combination of shaped lenses (e.g., fisheye lenses; see Fig. 1a) and can reach a field of view even bigger than 180° (i.e., slightly more than a hemisphere). Catadioptric cameras combine a standard camera with a shaped mirror – such as a parabolic, hyperbolic, or elliptical mirror – and provide 360-degree field of view in the horizontal plane and more than 100° in elevation. In Fig. 1b, you can see an example catadioptric camera using a hyperbolic mirror. Finally, polydioptric cameras use multiple cameras with overlapping field of view (Fig. 1c) and so far are the only cameras that provide a real omnidirectional (spherical) field of view (i.e., 4π steradians).

Catadioptric cameras were first introduced in robotics in 1990 by Yagi and Kawato [1], who used them for localizing robots. Fisheye cameras started to spread over only in 2000 thanks to new manufacturing techniques and precision tools that led to an increase of their field of view up to 180° and more. However, it is only since 2005 that these cameras have been miniaturized to the size of 1–2 cm and their field of view has been increased up to 190° or even more (see, for instance, Fig. 2a).

In the next sections, an overview on omnidirectional camera models and calibration will be given. For an in-depth study on omnidirectional vision, the reader is referred to [2–4] and to [5] for a more detailed survey on omnidirectional camera models.



Omnidirectional Camera, Fig. 1 (**a**) Dioptric camera (e.g., fisheye); (**b**) catadioptric camera; (**c**) an example polydioptric camera produced by Immersive Media



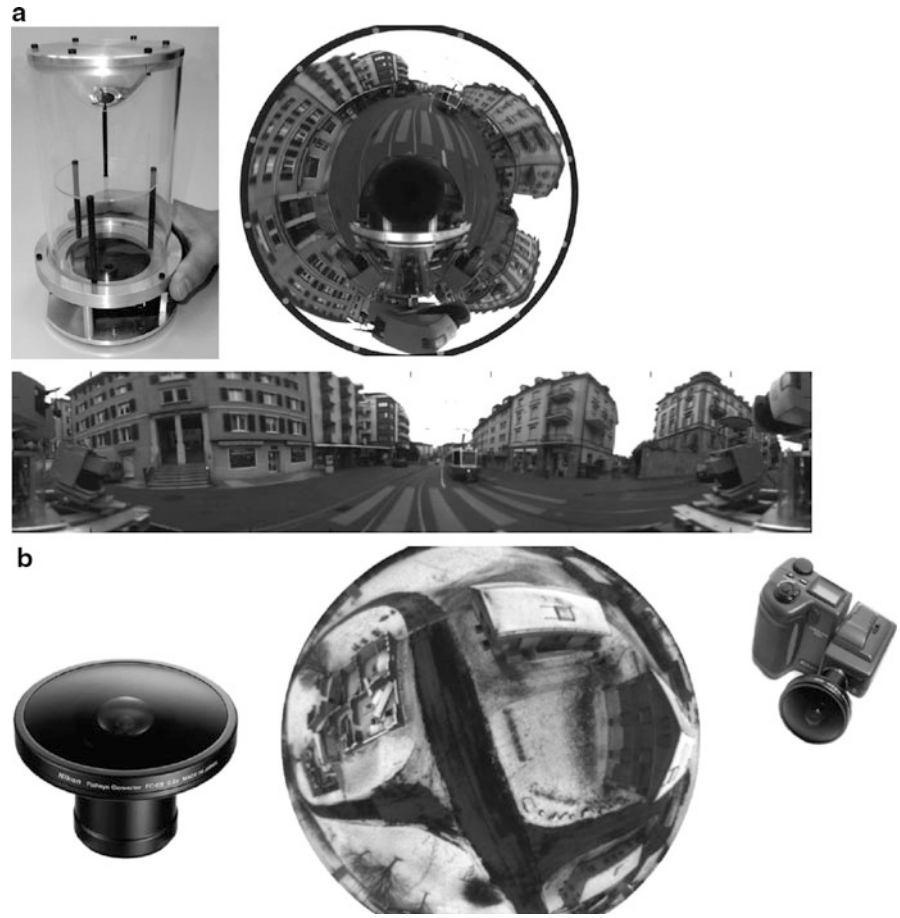
Omnidirectional Camera, Fig. 2 (**a**) The fisheye lens from Omnitech Robotics (www.omnitech.com) provides a field of view of 190°. This lens has a diameter of 1.7 cm. This camera has been used on the sFly autonomous helicopter at the ETH Zurich, [18]. (**b**) A miniature catadioptric camera built at the ETH Zurich, which is also used for autonomous flight. It uses a spherical mirror and a transparent plastic support. The camera measures 2 cm in diameter and 8 cm in height. (**c**) The muFly camera built by CSEM, which is used on the muFly helicopter at the ETH Zurich. This is one of the smallest catadioptric cameras ever built. Additionally, it uses a polar CCD (**d**) where pixels are arranged radially

Theory

Central Omnidirectional Cameras

A vision system is said to be central when the optical rays to the viewed objects intersect in a single point in 3D called projection center or single effective viewpoint (Fig. 4). This property is called single effective viewpoint property. The perspective camera is an example of a central projection system because all optical rays intersect in one point, that is, the camera optical center.

All modern fisheye cameras are central, and hence, they satisfy the single effective viewpoint property. Central catadioptric cameras conversely can be built only by opportunely choosing the mirror shape and the distance between the camera and the mirror. As proven by Baker and Nayar [6], the family of mirrors that satisfy the single viewpoint property is the class

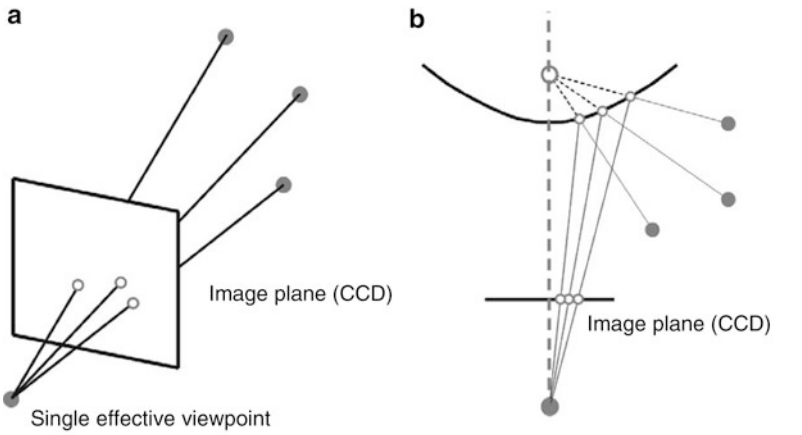


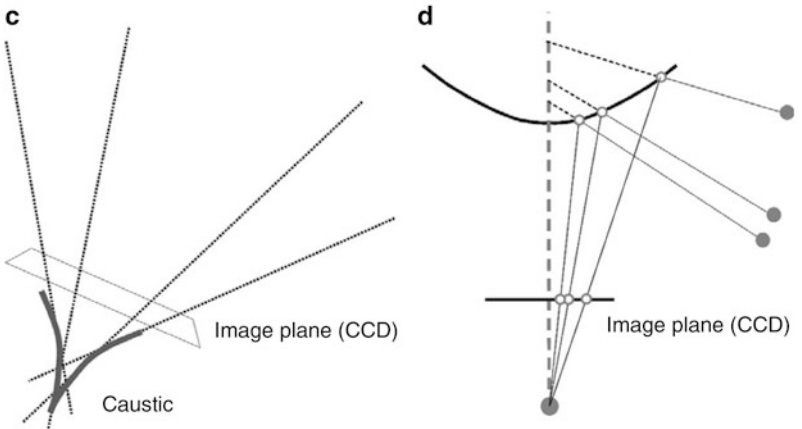
Omnidirectional Camera, Fig. 3 (**a**) A catadioptric omnidirectional camera using a hyperbolic mirror. The image is typically unwrapped into a cylindrical panorama. The field of view is typically 100° in elevation and 360° in azimuth. (**b**) Nikon fisheye lens FC-E8. This lens provides a hemispherical (180°) field of view

of rotated (swept) conic sections, that is, hyperbolic, parabolic, and elliptical mirrors. In the case of hyperbolic and elliptical mirrors, the single view point property is achieved by ensuring that the camera center (i.e., the pinhole or the center of the lens) coincides with one of the foci of the hyperbola (ellipse) (Fig. 5). In the case of parabolic mirrors, an orthographic lens must be interposed between the camera and the mirror; this makes it possible that parallel rays reflected by the parabolic mirror converge to the camera center (Fig. 5).

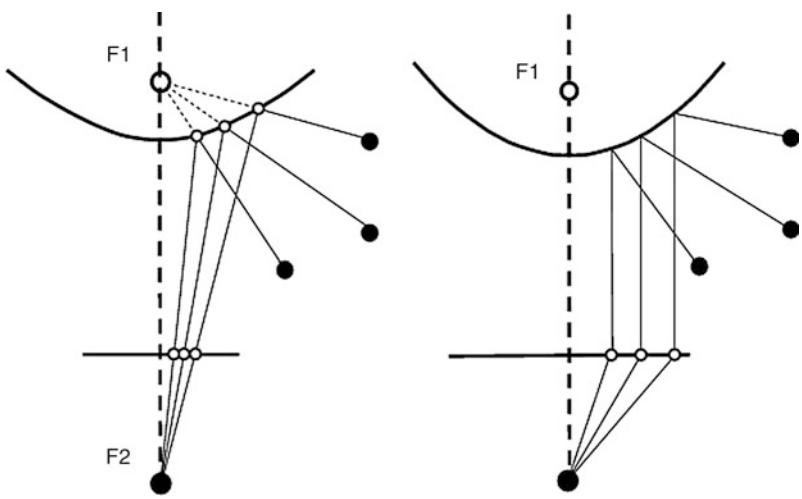
The reason a single effective viewpoint is so desirable is that it allows the user to generate geometrically correct perspective images from the pictures captured by the omnidirectional camera (Fig. 6). This is possible because, under the single view point constraint, every pixel in the sensed image measures the irradiance of the light passing through the viewpoint in one particular direction. When the geometry of the omnidirectional camera is known, that is, when the camera is calibrated, one can precompute this direction for each pixel. Therefore, the irradiance value measured by each pixel can be mapped onto a plane at any distance from the viewpoint to form a planar perspective image. Additionally, the image can be mapped on to a sphere centered on the single viewpoint, that is, spherical projection (Fig. 6, bottom).

Another reason why the single view point property is so important is that it allows the user to apply the well-known theory of epipolar geometry, which is extremely important for structure from motion. Epipolar geometry holds for any central camera, both perspective and omnidirectional.





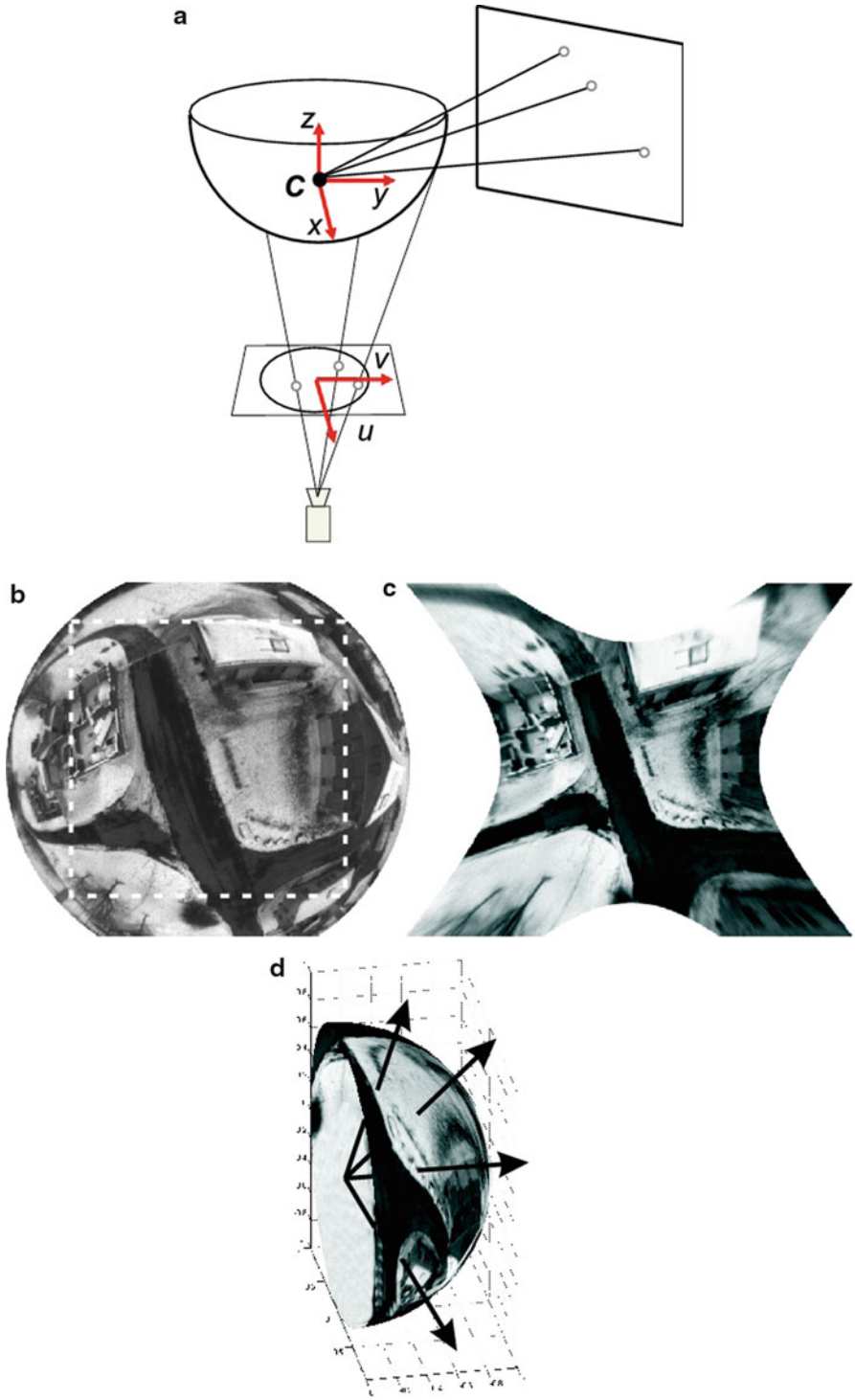
Omnidirectional Camera, Fig. 4 (**a**) and (**b**) Example of central cameras: perspective projection and catadioptric projection through a hyperbolic mirror. (**c**) and (**d**) Example of noncentral cameras: the envelope of the optical rays forms a caustic



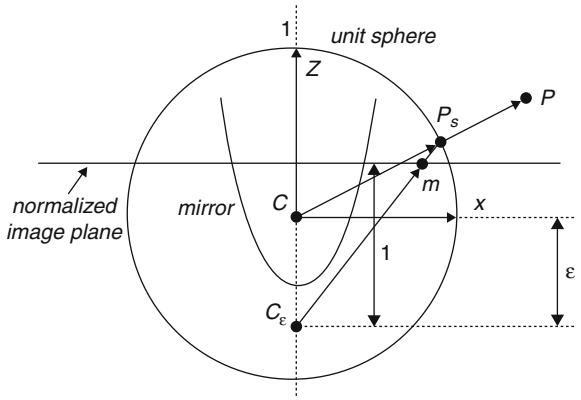
Omnidirectional Camera, Fig. 5 Central catadioptric cameras can be built by using hyperbolic and parabolic mirrors. The parabolic mirror requires the use of an orthographic lens

Omnidirectional Camera Model and Calibration

Intuitively, the model of an omnidirectional camera is a little more complicated than a standard perspective camera. The model should indeed take into account the reflection operated by the mirror in the case of a catadioptric camera or the refraction caused by the lens in the case of a fisheye camera. Because the literature in this field is quite large, this entry reviews two different projection models that have become standards in omnidirectional vision and robotics. Additionally,



Omnidirectional Camera, Fig. 6 Central cameras allow the user to remap regions of the omnidirectional image into a perspective image. This can be done straightforwardly by intersecting the optical rays with a plane specified arbitrarily by the user (**a**). For obvious reasons, we cannot project the whole omnidirectional image onto a plane but only subregions of it (**b-c**). Another alternative is the projection onto a sphere (**d**). In this case, the entire omnidirectional image can be remaped to a sphere



Omnidirectional Camera, Fig. 7 Unified projection model for central catadioptric cameras of Geyer and Daniilidis

Matlab toolboxes have been developed for these two models, which are used worldwide by both specialists and nonexperts.

The first model is known as the unified projection model for central catadioptric cameras. It was developed in 2000 by Geyer and Daniilidis [7] (later refined by Barreto and Araujo [8]), who have the merit of having proposed a model that encompasses all three types of central catadioptric cameras, that is, cameras using a hyperbolic, parabolic, or elliptical mirror. This model was developed specifically for central catadioptric cameras and is not valid for fisheye cameras. The approximation of a fisheye lens model by a catadioptric one is usually possible – however, with limited accuracy only – as investigated in [9].

Conversely, the second model unifies both central catadioptric cameras and fisheye cameras under a general model also known as Taylor model. It was developed in 2006 by Scaramuzza et al. [10, 11] and has the advantage that both catadioptric and dioptric cameras can be described through the same model, namely, a Taylor polynomial.

Unified Model for Central Catadioptric Cameras

With their landmark paper from 2000, Geyer and Daniilidis showed that every catadioptric (parabolic, hyperbolic, elliptical) and standard perspective projection is equivalent to a projective mapping from a sphere, centered in the single viewpoint, to a plane with the projection center on the perpendicular to the plane and distant ϵ from the center of the sphere. This is summarized in Fig. 7.

Omnidirectional Camera, Table 1 ϵ values for different types of mirrors

Mirror type	ϵ
Parabola	1
Hyperbola	$\frac{d}{\sqrt{d^2+4l^2}}$
Ellipse	$\frac{d}{\sqrt{d^2+4l^2}}$
Perspective	0

The goal of this section is to find the relation between the viewing direction to the scene point and the pixel coordinates of its corresponding image point. The projection model of Geyer and Daniilidis follows a four-step process. Let $P = (x, y, z)$ be a scene point in the mirror reference frame centered in C (Fig. 7). For convenience, we assume that the axis of symmetry of the mirror is perfectly aligned with the optical axis of the camera. We also assume that the x and y axes of the camera and mirror are aligned. Therefore, the camera and mirror reference frames differ only by a translation along z.

– The first step consists in projecting the scene point onto the unit sphere; therefore,

$$P_s = \frac{P}{\|P\|} = (x_s, y_s, z_s). \tag{1}$$

– The point coordinates are then changed to a new reference frame centered in $C_\epsilon = (0, 0, -\epsilon)$; therefore,

$$P_\epsilon = (x_s, y_s, z_s + \epsilon). \tag{2}$$

Observe that ϵ ranges between 0 (planar mirror) and 1 (parabolic mirror). The correct value of ϵ can be obtained knowing the distance d between the foci of the conic and the latus rectum l as summarized in Table 1. The latus rectum of a conic section is the chord through a focus parallel to the conic section directrix.

– P_ϵ is then projected onto the normalized image plane distant 1 from C_ϵ; therefore,

$$\tilde{m} = (x_m, y_m, 1) = \left(\frac{x_s}{z_s + \epsilon}, \frac{y_s}{z_s + \epsilon}, 1\right) = g^{-1}(P_s). \tag{3}$$

– Finally, the point $\tilde{m}$ is mapped to the camera image point $\tilde{p} = (u, v, 1)$ through the intrinsic parameter

matrix K; therefore,

$$\tilde{p} = K\tilde{m}, \tag{4}$$

where K is

$$K = \begin{bmatrix} \alpha_u & \alpha_u \cot(\theta) & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5}$$

– It is easy to show that function g^{-1} is bijective and that its inverse g is given by:

$$P_s = g(m) \propto \begin{bmatrix} x_m \\ y_m \\ 1 - \epsilon \frac{x_m^2 + y_m^2 + 1}{\epsilon + \sqrt{1 + (1 - \epsilon^2)(x_m^2 + y_m^2)}} \end{bmatrix}, \tag{6}$$

where $\propto$ indicates that g is proportional to the quantity on the right-hand side. To obtain the normalization factor, it is sufficient to normalize $g(m)$ onto the unit sphere.

Equation (6) can be obtained by inverting (3) and imposing the constraint that P_s must lie on the unit sphere and, thus, $x_s^2 + y_s^2 + z_s^2 = 1$. From this constraint, we then get an expression for z_s as a function of ϵ, x_m, and y_m. More details can be found in [12].

Observe that Eq. (6) is the core of the projection model of central catadioptric cameras. It expresses the relation between the point m on the normalized image plane and the unit vector P_s in the mirror reference frame. Note that in the case of planar mirror, we have $\epsilon = 0$ and (6) becomes the projection equation of perspective cameras $P_s \propto (x_m, y_m, 1)$.

This model has proved to be able to describe accurately all central catadioptric cameras (parabolic, hyperbolic, and elliptical mirror) and standard perspective cameras. An extension of this model for fisheye lenses was proposed in 2004 by Ying and Hu [9]. However, the approximation of a fisheye camera through a catadioptric one works only with limited accuracy. This is mainly because, while the three types of central catadioptric cameras can be represented through an exact parametric function (parabola, hyperbola, ellipse), the projective models of fisheye lenses vary from camera to camera and depend on the lens field of view. To overcome this problem, a new unified model was proposed, which will be described in the next section.

Unified Model for Catadioptric and Fisheye Cameras

This unified model was proposed by Scaramuzza et al. in 2006 [10, 11]. The main difference with the previous model lies in the choice of the function g. To overcome the lack of knowledge of a parametric model for fisheye cameras, the authors proposed the use of a Taylor polynomial, whose coefficients and degree are found through the calibration process. Accordingly, the relation between the normalized image point $\tilde{m} = (x_m, y_m, 1)$ and the unit vector P_s in the fisheye (mirror) reference frame can be written as:

$$P_s = g(m) \propto \begin{bmatrix} x_m \\ y_m \\ a_0 + a_2\rho^2 + \ldots + a_N\rho^N \end{bmatrix}, \tag{7}$$

where $\rho = \sqrt{x_m^2 + y_m^2}$. As the reader may observe, the first-order term (i.e., $a_1\rho$) of the polynomial is missing. This follows from the observation that the first derivative of the polynomial calculated at $\rho = 0$ must be null for both catadioptric and fisheye cameras (this is straightforward to verify for catadioptric cameras by differentiating (6)). Also observe that because of its polynomial nature, this expression can encompass catadioptric, fisheye, and perspective cameras. This can be done by opportunely choosing the degree of the polynomial. As highlighted by the authors, polynomials of order 3 or 4 are able to model very accurately all catadioptric cameras and many types of fisheye cameras available on the market. The applicability of this model to a wide range of commercial cameras is at the origin of its success.

Omnidirectional Camera Calibration

The calibration of omnidirectional cameras is similar to that for calibrating standard perspective cameras. Again, the most popular methods take advantage of planar grids that are shown by the user at different positions and orientations. For omnidirectional cameras, it is very important that the calibration images are taken all around the camera and not on a single side only. This is in order to compensate for possible misalignments between the camera and mirror.

It is worth to mention three open-source calibration toolboxes currently available for Matlab, which differ mainly for the projection model adopted and the type of calibration pattern:

- The toolbox of Mei uses checkerboard-like images and takes advantage of the projection model of Geyer and Daniilidis discussed earlier. It is particularly suitable for catadioptric cameras using hyperbolic, parabolic, folded mirrors, and spherical mirrors. Mei's toolbox can be downloaded from [13], while the theoretical details can be found in [14].
- The toolbox of Barreto uses line images instead of checkerboards. Like the previous toolbox, it also uses the projection model of Geyer and Daniilidis. It is particularly suitable for parabolic mirrors. The toolbox can be downloaded from [12], while the theoretical details can be found in [15] and [16].
- Finally, the toolbox of Scaramuzza uses checkerboard-like images. Contrary to the previous two, it takes advantage of the unified Taylor model for catadioptric and fisheye cameras developed by the same author. It works with catadioptric cameras using hyperbolic, parabolic, folded mirrors, spherical, and elliptical mirrors. Additionally, it works with a wide range of fisheye lenses available on the market – such as Nikon, Sigma, and Omnitech Robotics – with field of view up to 195°. The toolbox can be downloaded from [17], while the theoretical details can be found in [10] and [11]. Contrary to the previous two toolboxes, this toolbox features an automatic calibration process. In fact, both the center of distortion and the calibration points are detected automatically without any user intervention. This toolbox became very popular and is currently used at several companies such as NASA, Philips, Bosch, Daimler, and XSens.

Application

Thanks to the camera miniaturization, to the recent developments in optics manufacturing, and to the decreasing prices in the camerasï£¡ market, catadioptric and dioptric omnidirectional cameras are being more and more used in different research fields. Miniature dioptric and catadioptric cameras are now used by the automobile industry in addition to sonars for improving safety, by providing to the driver an omnidirectional view of the surrounding environment. Miniature fisheye cameras are used in endoscopes for surgical operations or onboard microaerial vehicles for pipeline inspection as well as rescue operations. Other examples involve meteorology for sky observation.

Roboticists have also been using omnidirectional cameras with very successful results on robot localization, mapping, and aerial and ground robot navigation [18–23]. Omnidirectional vision allows the robot to recognize places more easily than with standard perspective cameras [24]. Furthermore, landmarks can be tracked in all directions and over longer periods of time, making it possible to estimate motion and build maps of the environment with better accuracy than with standard cameras; see Fig. 2 for some of examples of miniature omnidirectional cameras used on state-of-the-art micro aerial vehicles. Several companies, like Google, are using omnidirectional cameras to build photorealistic street views and three-dimensional reconstructions of cities along with texture. Two example omnidirectional images are shown in Fig. 3.

References

1. Yagi Y, Kawato S (1990) Panorama scene analysis with conic projection. In: IEEE international conference on intelligent robots and systems, workshop on towards a new frontier of applications, Ibaraki
2. Benosman R, Kang, S (2001) Panoramic vision: sensors, theory, and applications. Springer, New York
3. Daniillidis K, Klette R (2006) Imaging beyond the pinhole camera. Springer, New York
4. Scaramuzza D (2008) Omnidirectional vision: from calibration to robot motion estimation, PhD thesis n. 17635, ETH Zurich
5. Sturm P, Ramalingam S, Tardif J, Gasparini S, Barreto J (2010) Camera models and fundamental concepts used in geometric computer vision. FInt J Comput Vis 35(2):175–196
6. Baker S, Nayar S, A theory of single-viewpoint catadioptric image formation. Int J Comput Vis 35(2):175–196
7. Geyer C, Daniilidis K (2000) A unifying theory for central panoramic systems and practical applications. In: European conference on computer vision (ECCV), Dublin
8. Barreto J (2001) Issues on the geometry of central catadioptric image formation, Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition (CVPR)
9. Ying X, Hu Z (2004) Can we consider central catadioptric cameras and fisheye cameras within a unified imaging model? In: European conference on computer vision (ECCV). Lecture notes in computer science. Springer, Berlin
10. Scaramuzza D, Martinelli A, Siegwart R (2006) A flexible technique for accurate omnidirectional camera calibration and structure from motion. In: IEEE international conference on computer vision systems, New York
11. Scaramuzza D, Martinelli A, Siegwart R (2006) A toolbox for easy calibrating omnidirectional cameras. In: IEEE international conference on intelligent robots and systems, Beijing

12. Barreto J, Omnidirectional camera calibration toolbox for matlab (ocamcalib toolbox). http://www.isr.uc.pt/~jpbar/CatPack/pag1.htm
13. Mei C, Omnidirectional camera calibration toolbox for matlab. http://homepages.laas.fr/~cmei/index.php/Toolbox
14. Mei C, Rives P(2007) Single view point omnidirectional camera calibration from planar grids. In: IEEE international conference on robotics and automation, Roma
15. Barreto J, Araujo H (2005) Geometric properties of central catadioptric line images and their application in calibration. IEEE Trans Pattern Anal Mach Intell 27(8): 1237–1333
16. Barreto J, Araujo H (2006) Fitting conics to paracatadioptric projection of lines. Comput Vis Image Underst 101(3): 151–165
17. Scaramuzza D, Omnidirectional camera calibration toolbox for matlab (ocamcalib toolbox) Google for "*ocamcalib*". https://sites.google.com/site/scarabotix/ocamcalib-toolbox
18. Blosch M, Weiss S, Scaramuzza D, Siegwart R (2010) Vision based mav navigation in unknown and unstructured environments. IEEE international conference on robotics and automation, Anchorage
19. Bosse M, Rikoski R, Leonard J, Teller S (2002) Vanishing points and 3d lines from omnidirectional video. In: International conference on image processing, Rochester
20. Corke P, Strelow D, Singh S (2004) Omnidirectional visual odometry for a planetary rover. In: IEEE/RSJ international conference on intelligent robots and systems, Sendai
21. Scaramuzza D, Siegwart R (2008) Appearance-guided monocular omnidirectional visual odometry for outdoor ground vehicles. IEEE Trans Robot 24(5): 1015–1026
22. Scaramuzza D, Fraundorfer F, Siegwart R (2009) Real-time monocular visual odometry for on-road vehicles with 1-point ransac. In: IEEE international conference on robotics and automation, Kobe
23. Tardif J, Pavlidis Y, Daniilidis K (2008) Monocular visual odometry in urban environments using an omnidirectional camera. In: IEEE/RSJ international conference on intelligent robots and systems, Nice
24. Scaramuzza D, Fraundorfer F, Pollefeys M (2010) Closing the loop in appearance-guided omnidirectional visual odometry by using vocabulary trees. Robot Auton Syst J. 58(6):820–827. Elsevier

Omnidirectional Vision

Peter Sturm
INRIA Grenoble Rhône-Alpes, St Ismier Cedex, France

Related Concepts

►Field of View; ►Fisheye Lens

Definition

Omnidirectional vision consists of the theoretical and practical computer vision approaches devised for images acquired by omnidirectional cameras, that is, cameras with a very large field of view, typically hemispherical or larger.

Background

Most regular cameras have a restricted field of view, typically up to several tens of degrees. Omnidirectional cameras, on the contrary, have hemispheric or even larger, up to complete spherical, fields of view. Computer vision theories and algorithms dedicated to omnidirectional images are subsumed under the expression omnidirectional vision. Researchers working on omnidirectional vision have also proposed novel designs of omnidirectional cameras.

The interest of building omnidirectional cameras arose rather soon after the invention of photography, with the first known panoramic camera built in 1843 [1]. To be precise, this and other early panoramic cameras had a restricted vertical field of view and are thus not truly omnidirectional.

There exist several technologies to acquire omnidirectional images. The first to be used was based on rotating a regular camera about its optical center and optically or computationally stitching images together to a panoramic image. *Panoramic cameras* can be considered as a subset of omnidirectional ones, in that they usually deliver an extended field of view only in one direction. If the camera is rotated about different axes, acquired images may be stitched to form a complete omnidirectional mosaic.

Probably the second omnidirectional technology was based on multi-camera systems, where the rotating camera was replaced by several colocated cameras with overlapping fields of view. Such systems were built at least as early as in 1884 [2, 3].

The most common single-camera solutions that allow the acquisition of instantaneous omnidirectional images, that is, without having to acquire multiple images while rotating a camera, are *fisheye cameras* and *catadioptric cameras*. The latter achieve large fields of view by having one or more cameras look at a curved mirror or one or more planar or curved mirrors.

Theory

When working with images acquired by omnidirectional cameras, one usually first needs an *image formation model*. Many camera models have been proposed for omnidirectional systems. Some of them resemble classical models for radial and tangential distortion in that they are based on polynomial expressions. Others, typically models for fisheye lenses, use trigonometric functions and yet others are specific to catadioptric cameras. A recent overview of such camera models can be found in [3]. Common to all models is that they allow to project 3D points onto image points and/or perform back projection, from image points to lines of sight in 3D.

Besides purely geometrical aspects of omnidirectional image formation, other aspects and properties have been examined such as focusing mechanisms, blur, and chromatic aberrations; see, for instance, [4]. Generally speaking, these issues all tend to be more complex than with regular cameras.

The *calibration* of an omnidirectional camera is in principle not different from that of a regular camera, in that it exploits images of reference objects with known shape or special properties, for example, showing straight lines on their surface. In practice, calibrating an omnidirectional camera is usually more complex, since usual calibration objects do not fill the field of view completely, thus requiring the acquisition of more images. Also, omnidirectional images are often less sharp and are by definition less resolved, thus making feature extraction and matching more challenging.

Besides modeling and calibrating omnidirectional cameras, other fundamental works in omnidirectional vision are dedicated to image processing. The goal of these works is to adapt image processing operations, for example, for edge or interest point extraction, to the image formation model which usually includes strong distortions, in order to enhance their performance [5, 6].

Application

Once an omnidirectional camera is calibrated, one can in principle use the same algorithms for tasks such as pose estimation, motion estimation, or 3D modeling as for regular cameras and the usual model thereof (pinhole model). There exist several differences though. For example, for motion estimation, results with omnidirectional cameras are often better than with cameras with a smaller field of view [7]. With a small field of view, a lateral translation and a lateral rotation give rise to similar optical flows, whereas in an omnidirectional image, the optical flows in the "sidewards" looking parts of the image will be very different for these two types of motion. This explains why motion estimation may be expected to be more stable when performed with omnidirectional images. For pose estimation, the converse is often the case, simply because the spatial resolution is lower than with a smaller field of view, decreasing the accuracy of the estimated object pose. Further, as mentioned above, processing of omnidirectional images may have to have recourse to specific approaches, in order to handle the strong distortions present in them.

Typical applications of omnidirectional vision are those where the extended field of view is directly beneficial. Among the first applications of omnidirectional images were the study of cloud formations in meteorology and the measurement of leaf coverage from fisheye images of forest canopies. A main application field for omnidirectional vision is the navigation of mobile robots, where obstacles can be detected instantaneously in all directions and where an omnidirectional camera allows for a more stable motion estimation and a more complete path planning than a narrow field of view camera. Other obvious usages are in video surveillance and tele-presence applications.

References

1. McBride B (2011) A timeline of panoramic cameras. http://www.panoramicphoto.com/timeline.htm. Accessed 3 Aug 2011
2. Tissandier G (1886) La photographie en ballon. Gauthier-Villars, Paris
3. Sturm P, Ramalingam S, Tardif JP, Gasparini S, Barreto J (2011) Camera models and fundamental concepts used in geometric computer vision. Found Trends Comput Graph Vis 6(1–2):1–183
4. Baker S, Nayar S (1999) A theory of single-viewpoint catadioptric image formation. Int J Comput Vis 35(2):1–22
5. Daniilidis K, Makadia A, Bülow T (2002) Image processing in catadioptric planes: spatiotemporal derivatives and optical flow computation. In: Proceedings of the workshop on omnidirectional vision, copenhagen, Denmark, pp 3–10
6. Bülow T (2002) Multiscale image processing on the sphere. In: Proceedings of the 24th DAGM symposium. Lecture notes in computer science, Zurich, Switzerland, vol 2449, pp 609–617

7. Nelson R, Aloimonos J (1988) Finding motion parameters from spherical motion fields (or the advantages of having eyes in the back of your head). Biol Cybern 58(4):261–273
8. Yagi Y (1999) Omnidirectional sensing and applications. IEICE Trans Inf Syst E82-D(3):568–579
9. Benosman R, Kang S (eds) (2001) Panoramic vision. Springer, New York

Opacity

Ivo Ihrke
MPI Informatik, Saarland University, Saarbrücken, Germany

Synonyms

Opaque

Related Concepts

►Polarization; ►Transparency and Translucency

Definition

Opacity describes the degree to which an object or material is not transmitting light. Sometimes the opposite is referred to as *transmittance* (mostly in physics). In computer graphics, the opposite of opacity is typically referred to as *transparency*.

Background

Most computer vision algorithms, whether designed for 3D reconstruction, object detection, object classification, etc., require the scene and the objects it consists of to be opaque. More precisely, objects are required to be diffusely opaque. Specularly reflective objects, e.g., are opaque by definition but most vision techniques cannot handle them.

A multitude of research has gone into lifting the restriction on scene opacity. When going beyond opaque objects, images are typically enhanced by an *alpha channel* – a fractional measure of opacity, or the amount by which an object obscures the background in an image.

In television and film production, the alpha channel is extensively used and usually estimated by *chroma keying* [1]. This technique uses a colored background in order to estimate the alpha channel.

Theory

The following contains a discussion of the physical (optical) principles causing the perception of opacity. Please refer to the classic textbook by Born and Wolf [2] for a detailed discussion.

Physical Processes

There is not a single physical process giving rise to what is known as opacity in computer vision. There can be a multitude of reasons for opacity to occur. The physical processes involved are those of *attenuation* and *scattering* of light upon molecular interaction with an object material, Fig. 1. A further, unrelated reason is physical *sensor integration* over the area of a single pixel: If the scene consists of many tiny objects that are somehow interacting with the surrounding light, seen against a background, sensor integration averages over the light contribution from the objects and the background. This can often be seen when observing thin structures like hair or fur, or leaves on distant trees.

Scattering describes a broad range of physical processes, both particle and wave phenomena, upon interaction of light with a material. The term itself describes deviation of light from a straight path due to inhomogeneities in the material that is being traversed. As an example, scattering can be caused by small particles in paints, droplets in clouds and water spray, soot particles in smoke, and many more. Waves can be scattered by obstacles that are small in comparison to the wavelength. A formal description of energy transfer in scattering media, of which light transport is a special case, can be found in [3].

Scattering can occur both in transmission and in reflection. The latter is usually described by the *bidirectional reflection distribution function* (BRDF), whereas the former is described by the *bidirectional transmission distribution function* (BTDF). These functions describe scattering at an interface or due to very thin transmissive structures. Sometimes, the combined effect of reflection and transmission at an interface is described by a single *bidirectional scattering*

distribution function (BSDF). This family of distribution functions can be seen as a statistical description of the scattering process: The probability that a particular photon is redirected into a given direction is modeled by the density function. A deterministic description, on the other hand, would involve quantum mechanical concepts (Fig. 1).

In contrast, for volumetric phenomena, the redistribution of light is described by the *scattering phase function*. For light traversing a volume of participating media, the major effects are out-scattering, i.e., light being deflected from its original, unobstructed path, and in-scattering, i.e., light being deflected from elsewhere onto the unobstructed path of a photon, Fig. 1. If only a single scattering event occurs, the process is called *single scattering*, otherwise it is referred to as *multiple scattering*.

The above mentioned effects all belong to the family of inelastic scattering processes where photons are essentially bounced around without being modified in wavelength. The counterpart, elastic scattering describes processes where a photon is absorbed by some atom just to be released in a different direction. This way, a change of energy levels can occur. The major effects are fluorescence and phosphorescence.

Attenuation is due to the physical process of absorption: Photons, while traversing some medium, may collide with molecules that are able to absorb their energy, Fig. 1. Usually this energy is converted to a faster motion of the molecule, producing heat as a net effect. The amount of absorption depends on the path length of the photon in the traversed medium. It is given by the

Beer-Lambert law

$$\frac{L}{L_0} = \exp^{-\int_{s_0}^{s_1} \sigma_s \circ c(s) + \sigma_a \circ c(s) ds}.$$

Here, L_0 is the incident radiance and L the radiance after attenuation. The path of the photon is described by the curve $c(s) : \mathbb{R} \mapsto \mathbb{R}^3$, σ_s is the *absorption coefficient* and σ_a the *scattering coefficient*. The scattering coefficient describes the loss of light due to out-scatter, while the absorption coefficient describes the loss due to absorption. Sometimes, the factor $\alpha = \sigma_s + \sigma_a$ is referred to as *attenuation coefficient*.

The linearized version of the Beer-Lambert law $\ln(L/L_0) = -\int_{s_0}^{s_1} \sigma_s \circ c(s) + \sigma_a \circ c(s) ds$ is the basis for emission and absorption tomography.

The *Fresnel Effect* is a natural source of attenuation and blending at interfaces of materials with different refractive index. A ray is usually split into a transmitted and a reflected part. While the geometry and the law of reflection, respective Snell's law, determine only the directions of the reflected and refracted ray, the radiance ratio can be computed with the Fresnel formulae. The transmittance and reflectance depend on the polarization state of the incident ray.

Refraction

As previously discussed, a ray of light crossing an interface is typically both reflected and transmitted. The transmitted part is usually refracted as well, see Fig. 1. This results in a different background pixel to be seen if a refractive object is moved into a ray that previously hit the background. A single opacity value for a camera pixel is thus not sufficient to explain the complex change in light transport. Rather, the opacity value must be associated with the ray in question. For opaque objects, the attenuation of the transmitted ray is very strong. They are thus perceived as not transmitting light. A special case of refraction occurs in unevenly heated media like gases or liquids or in mixtures of different gases. Here, *continuous refraction* is taking place, resulting in curved light paths. In wave optics, the *complex refractive index* is used as a mathematical tool to represent continuous refraction and attenuation in one number.

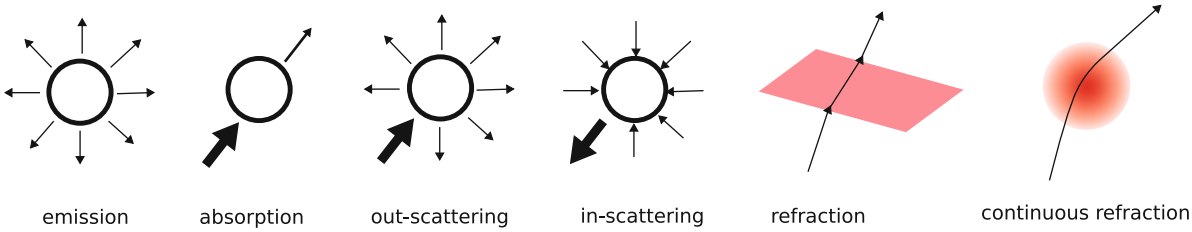
Application

Alpha Matting

Alpha matting is an extension to chroma keying discussed in the background section. Here, the more general problem where the background is arbitrary instead of consisting of a single color is considered. This problem comes in several flavors:

- Dynamic scene, static known background
- Static scene, static unknown background, i.e., natural image matting
- Dynamic scene, static unknown background
- Dynamic scene, dynamic background, i.e., video matting [4]

The corresponding process of synthetically overlaying an object image onto some background is known as *alpha blending*.



Opacity, Fig. 1 The major light transport effects in participating media (from *left* to *right*): emission, absorption, out-scattering, in-scattering, and two types of refraction

Apart from these two-dimensional applications, opacity has been used to three-dimensionally model transparent or translucent objects either phenomenologically as partially transparent view-dependent visual hulls [5] or for estimating the internal structure of volumetric light-emitting or absorbing objects and phenomena. The latter is usually performed using tomographic techniques [6].

Tomography

Tomography is the process of determining a function from its projections. In the tomographic sense, a projection is a line integral through some volumetric function. A collection of line integrals, taken from many different directions, can be used to compute the inner structure of volumetric phenomena. The best known application is computed tomography (CT). In computer vision, attempts to reconstruct volumetric phenomena like fire, smoke, and plasmas using tomographic techniques applied to observed opacities, even dynamically, have shown good success.

References

1. Gerald Millerson J Owens (2009) Television production. Focal Press, Amsterdam/Boston
2. Born M, Wolf E (1999) Principles of optics, 7th edn. Cambridge University Press, Cambridge, UK
3. Siegel R, Howell JR (1992) Thermal radiation heat transfer, 3rd edn. Taylor & Francis, New York, USA
4. Chuang YY, Agarwala A, Curless B, Salesin DH, Szeliski R (2002) Video matting of complex scenes. In: SIGGRAPH '02: proceedings of the 29th annual conference on computer graphics and interactive techniques, San Antonio. ACM, New York, pp 243–248
5. Matusik W, Pfister H, Ngan A, Beardsley P, Ziegler R, McMillan L (2002) Image-based 3D photography using opacity hulls. In: SIGGRAPH '02: proceedings of the 29th annual conference on computer graphics and interactive techniques, San Antonio. ACM, New York, pp 427–437
6. Ihrke I, Magnor M (2004) Image-based tomographic reconstruction of flames. In: ACM SIGGRAPH/Eurographics symposium proceedings, symposium on computer animation, Grenoble, pp 367–375

Opaque

▸Opacity

Optic Flow

▸Optical Flow

Optical Axis

Peter Sturm
INRIA Grenoble Rhône-Alpes, St Ismier Cedex, France

Synonyms

Principal axis

Related Concepts

▸Center of Projection; ▸Image Plane; ▸Pinhole Camera Model

Definition

For rotationally symmetric imaging systems, the optical axis is the (usually unique) axis of symmetry.

Background

The usual definition of optical axis in computer vision is tied to the pinhole camera model, where it is defined as the line passing through the center of projection and that is orthogonal to the image plane. For a perfectly aligned usual optical system and where the image plane is parallel to the lenses' principal planes, the optical axis is the line passing through the lenses' centers.

Theory

A more general definition is to consider the optical axis as the (unique) axis of rotational symmetry of an imaging system, if this symmetry exists. Here, it may be advisable to examine rotational symmetry of only the optical part of the image formation process and not the entire process from 3D to the pixel domain. This is meant to exclude the way the photosensitive surface samples the incoming light.

Let us examine this with the aid of the pinhole camera model as example. Suppose that the aspect ratio equals 1, that is, pixel density is identical in vertical and horizontal directions on the image plane. Then, the projection function from the 3D scene to 2D pixel coordinates is rotationally symmetric relative to the pinhole camera's optical axis and the image plane's principal point. However, if the aspect ratio deviates from 1, the rotational symmetry is broken. If one looks at what happens before light rays hit the image plane, then the projection function is still rotationally symmetric though.

A similar example is a camera that exhibits distortions due to the image plane not being perfectly parallel to the lenses' principal planes. Again, the full mapping from 3D to 2D pixel coordinates is not rotationally symmetric here.

To consider rotational symmetry and thus the existence of an optical axis, one may thus consider the function that maps light rays coming into the camera, to rays leaving the optics towards the image plane. In that respect, rotational symmetry and the concept of optical axis are also defined for noncentral cameras, that is, cameras without a single center of projection, like some catadioptric cameras.

References

1. Hecht E (2001) Optics, 4th edn. Addison Wesley, Reading
2. Geissler P (2000) Imaging optics. In: Jähne B, Haußecker H (eds) Computer vision and applications. Academic press, San Diego, pp 53–84

Optical Center

▸Center of Projection

Optical Flow

Thomas Brox
Department of Computer Science, University of Freiburg, Freiburg, Germany

Synonyms

Optic flow

Related Concepts

▸Rigid Registration

Definition

Optical flow is the vector field that describes the perceived motion of points in the image plane.

Background

When working with image sequences, analyzing the change of the image over time provides valuable information about the scene. The motion of points in the image, as described by the optical flow field, is an important cue in many computer vision tasks, such as tracking, motion segmentation, and structure-from-motion.

Estimating the optical flow from pairs of images is a classical computer vision task. The problem is approached by matching certain image structures, which are assumed to be more or less stable over

time, between two successive images. While earlier estimation methods could only provide sparse optical flow fields, nowadays it is common to estimate a dense optical flow field that provides a displacement vector for each pixel in the first image, describing where this pixel went in the second image.

Optical flow estimation is closely related to the problem of nonrigid registration. The main difference is that the latter problem often assumes a one-to-one mapping between two images. This assumption is not satisfied for optical flow due to occlusion.

Optical Flow Estimation

For estimating the optical flow field, a matching criterion is needed. In most cases this matching criterion is based on the pixel gray value or color, which leads to the so-called optical flow constraint:

$$I(x+u(x,y), y+v(x,y), t+1) - I(x,y,t) = 0, \quad (1)$$

where I denotes the image sequence and $w = (u,v)^\top$ the optical flow field. As this constraint is nonlinear in u and v, it is usually linearized to yield

$$I_x u + I_y v + I_t = 0, \quad (2)$$

where subscripts denote partial derivatives.

For each pixel, the optical flow constraint yields one equation. Since we have two unknowns per pixel to estimate, the problem is underconstrained without additional assumptions. This is also known as the aperture problem.

There are two solutions to this problem. One is to assume a parametric flow model in a fixed neighborhood around each pixel. Choosing the neighborhood (the aperture) large enough, one collects enough information to estimate the model parameters, which determine the flow. This methodology was initially proposed by Lucas and Kanade [1].

The other solution imposes a regularity constraint on the resulting flow field. Rather than estimating flow vectors for each pixel independently, this leads to a global estimation of the whole flow field and is expressed as an energy minimization problem. The first such model has been introduced by Horn and Schunck [2] and reads:

$$E(u,v) = \int (I_x u + I_y v + I_t)^2 + \alpha \left(|\nabla u|^2 + |\nabla v|^2 \right) dx dy. \quad (3)$$

The energy is minimized via variational methods. Since it is a convex functional, the global minimizer $(u,v)^\top$ can be found with standard techniques such as gradient descent or by solving the linear system emerging from the Euler-Lagrange equations.

Numerous extensions of the basic Horn-Schunck model have been presented over the years to deal with several shortcomings of the original model. One important modification is to replace the quadratic penalizers in (Eq. 3) by nonquadratic ones [3, 4]. This can be interpreted as replacing the inherent Gaussian distribution model in (Eq. 3) by one based on robust statistics. Long-tailed distributions allow for outliers in both the optical flow constraint and the smoothness assumption, i.e., occlusion and motion discontinuities are integrated into the model. The most popular choice to date is the use of a regularized Laplace distribution, which leads to the l_1-norm in the energy functional [5]. As the l_1-norm is the limit between convex and nonconvex penalizers, it allows for the maximum number of outliers while still leading to a convex optimization problem.

The linearization of the optical flow constraint in (Eq. 2) helps solving for the optical flow field, but it includes the assumption that the image is locally linear. In practice this is only true for very small neighborhoods with a few pixels radius and restricts the magnitude of the displacement vectors that can be estimated. In [6] it was suggested to work with the original, nonlinearized optical flow constraint and to postpone the linearization to the numerical scheme. As due to the nonlinearized optical flow constraint the energy becomes nonconvex, local minima are an issue and need to be approached by appropriate heuristics. In [5] this was shown to coincide with the ideas of earlier multiresolution approaches [1, 4]. The numerical scheme in [5] can be regarded as a Gauss-Newton method combined with a continuation strategy to avoid local minima.

Contemporary variational methods for optical flow estimation are based on the following energy functional:

$$E(u,v)=\int \Psi\left(I(x+u,y+v,t+1)-I(x,y,t))^2\right) + \alpha\Psi\left(|\nabla u|^2+|\nabla v|^2\right)\,dxdy, \quad (4)$$

where $\Psi(s^2)$ denotes a robust function, which is often chosen as the regularized l_1-norm $\Psi(s^2)=\sqrt{s^2+\epsilon^2}$ with a reasonably small regularization constant ϵ.

Other improvements include extended matching criteria that are invariant to brightness changes [7] and orientation-specific, anisotropic, or nonlocal regularizers [8, 9]. Multigrid solvers and efficient GPU implementations have pushed even advanced optical flow estimation techniques towards real-time performance [10, 11].

Instead of these variational techniques, one can approach optical flow estimation also with combinatorial methods. The most straightforward version of this is simple block matching, which leads to unsatisfactory results though. More enhanced combinatorial methods, e.g., based on belief propagation, perform much better [12, 13].

Nonetheless, variational techniques are currently the dominating way to estimate optical flow. This is very much in contrast to the related task of disparity estimation, where combinatorial methods are much more powerful. This is mainly because disparity estimation is a search problem among an ordered set of labels, which allows for efficient, globally optimal algorithms. In contrast, the two-dimensional nature of optical flow estimation leads to combinatorial problems that are NP-hard. Without the advantage of global optimality, pure combinatorial techniques are inferior to variational techniques due to discretization artifacts and missing subpixel accuracy. Most combinatorial approaches alleviate these problems by running a variational approach as a postprocessing step. Combinatorial techniques are promising in the sense that they can potentially better deal with large motion as well as occlusion. Brox and Malik [14] is an example of combining a combinatorial search (simple block matching) with variational methods to estimate faster motion.

Application

Typical applications of optical flow are tracking, motion segmentation, and action recognition. Tracking has again many applications in fields like structure-from-motion and surveillance. The most widely spread tracker, the KLT tracker [15], is based on the Lucas-Kanade technique described above. Trackers based on variational optical flow have been proposed as well [16, 17]. In motion segmentation, one can directly cluster the optical flow vectors, as done in the motion layer framework by Wang and Adelson [18]. A joint estimation of optical flow and corresponding motion segments has been proposed in [19, 20]. Histograms of the optical flow can be used for action recognition [21].

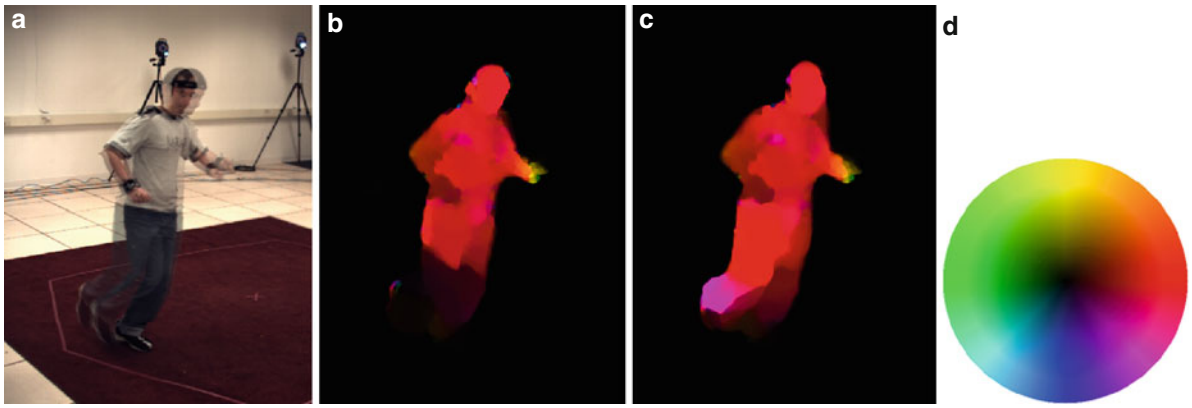
Open Problems

The main remaining problems in optical flow estimation are concerned with occlusion and precise motion boundary estimation, especially in homogenous image areas. While both are implicitly covered by state-of-the-art techniques in areas with rich structure, homogenous image areas yield the data term to be irrelevant and the estimation results are solely driven by the smoothness assumption, which is a relatively weak prior for such situations.

Experimental Results

A quantitative comparison of a large number of optical flow estimation methods is available at [22]. However, the numbers must be treated with care. As they are computed on the basis of only eight image pairs, overfitting is an issue. More complex methods with many partially hidden parameters have a clear advantage, as more parameters can be specifically adapted to the needs of the benchmark. For this reason, the top-performing methods in the benchmark are not necessarily the best performing ones on a more diverse set of natural videos. Nevertheless, the benchmark allows to coarsely separate techniques that lack certain qualities in the lower part of the table. The computation times, even though they are not directly comparable due to the usage of different hardware, also give hints on the applicability of techniques in practice.

Measuring the quality of optical flow estimation methods is generally hard, because deriving ground truth flow in real videos comes with a very big effort. With the improved rendering quality in computer graphics, large sets of realistic, synthetic test



Optical Flow, Fig. 1 From *left* to *right*: (**a**) Two-frame overlay showing large motion especially at the foot of the person. (**b**) Optical flow estimated with [5]. The motion of the upper body part is estimated well, but the motion of the leg is missed. (**c**) Optical flow estimated with large displacement optical flow [14]. The fast motion can be captured. (**d**) *Color code* used to represent the optical flow. For instance, *red* stands for motion to the right

sequences could be a possible way out of this dilemma and will ideally lead to larger benchmark datasets that better cover the space of scenes faced in optical flow applications.

Special properties of certain optical flow estimation techniques can be observed also qualitatively. Figure 1 compares a method that can explicitly deal with larger motion to one that does not have this specific capability.

References

1. Lucas B, Kanade T (1981) An iterative image registration technique with an application to stereo vision. In: Proceedings of the seventh international joint conference on artificial intelligence, Vancouver, pp 674–679
2. Horn B, Schunck B (1981) Determining optical flow. Artif Intell 17:185–203
3. Black MJ, Anandan P (1996) The robust estimation of multiple motions: parametric and piecewise smooth flow fields. Comput Vis Image Underst 63(1):75–104
4. Mémin E, Pérez P (1998) Dense estimation and object-based segmentation of the optical flow with robust techniques. IEEE Trans Image Process 7(5):703–719
5. Brox T, Bruhn A, Papenberg N, Weickert J (2004) High accuracy optical flow estimation based on a theory for warping. In: European conference on computer vision (ECCV). Volume 3024 of LNCS. Springer, Berlin/New York, pp 25–36
6. Alvarez L, Weickert J, Sánchez J (2000) Reliable estimation of dense optical flow fields with large displacements. Int J Comput Vis 39(1):41–56
7. Papenberg N, Bruhn A, Brox T, Didas S, Weickert J (2006) Highly accurate optic flow computation with theoretically justified warping. Int J Comput Vis 67:141–158
8. Zimmer H, Bruhn A, Weickert J, Valgaerts L, Salgado A, Rosenhahn B, Seidel HP (2009) Complementary optic flow. In: Proceedings of the 7th international conference on energy minimization methods in computer vision and pattern recognition. Volume 5681 of LNCS. Springer, Berlin/Heidelberg/New York, pp 207–220
9. Werlberger M, Pock T, Bischof H (2010) Motion estimation with non-local total variation regularization. In: International conference on computer vision and pattern recognition, San Francisco
10. Bruhn A (2006) Variational optic flow computation: accurate modelling and efficient numerics. PhD thesis, Faculty of Mathematics and Computer Science, Saarland University, Germany
11. Zach C, Pock T, Bischof H (2007) A duality based approach for realtime TV-L1 optical flow. In: Pattern recognition – proceeding DAGM. Volume 4713 of LNCS. Springer, Heidelberg pp 214–223
12. Shekhovtsov A, Kovtun I, Hlaváč VV (2007) Efficient MRF deformation model for non-rigid image matching. In: International conference on computer vision and pattern recognition (CVPR), Minneapolis
13. Glocker B, Paragios N, Komodakis N, Tziritas G, Navab N (2008) Optical flow estimation with uncertainties through dynamic MRFs. In: International conference on computer vision and pattern recognition (CVPR), Anchorage
14. Brox T, Malik J (2011) Large displacement optical flow: descriptor matching in variational motion estimation. In: IEEE transactions on pattern analysis and machine intelligence 33(3):500–513
15. Shi J, Tomasi C (1994) Good features to track. In: International conference on computer vision and pattern recognition (CVPR), Seattle, pp 593–600
16. Sand P, Teller S (2008) Particle video: long-range motion estimation using point trajectories. Int J Comput Vis 80(1):72–91
17. Sundaram N, Brox T, Keutzer K (2010) Dense point trajectories by GPU-accelerated large displacement optical

flow. In: European conference on computer vision (ECCV). LNCS. Springer, Berlin/New York
18. Wang JYA, Adelson EH (1994) Representing moving images with layers. IEEE Trans Image Process 3(5): 625–638
19. Weiss Y (1997) Smoothness in layers: motion segmentation using nonparametric mixture estimation. In: International conference on computer vision and pattern recognition (CVPR), San Juan, Puerto Rico, pp 520–527
20. Cremers D, Soatto S (2005) Motion competition: a variational framework for piecewise parametric motion segmentation. Int J Comput Vis 62(3):249–265
21. Efros A, Berg A, Mori G, Malik J (2003) Recognizing action at a distance. In: International conference on computer vision, Nice, pp 726–733
22. Middlebury optical flow benchmark. vision.middlebury.edu

Optimal Estimation

Kenichi Kanatani
Professor Emeritus of Okayama University,
Okayama, Japan

Synonyms

Optimal parameter estimation

Related Concepts

▸Maximum Likelihood Estimation

Definition

Optimal estimation in the computer vision context usually refers to estimating the parameters that describe the underlying problem from noisy observation. The estimation is done according to a given criterion of optimality, for which maximum likelihood is widely accepted. If Gaussian noise is assumed, it reduces to minimizing the Mahalanobis distance. If furthermore the Gaussian noise has a homogeneous and isotropic distribution, the procedure reduces to minimizing what is called the reprojection error.

Background

One of the central tasks of computer vision is the extraction of 2-D/3-D geometric information from noisy image data. Here, the term *image data* refers to values extracted from images by image processing operations such as edge filters and interest point detectors. Image data are said to be *noisy* in the sense that image processing operations for detecting them entail uncertainty to some extent.

For optimal estimation, a statistical model of observation needs to be introduced. Let $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$ be the observed image data. The standard model is to view each datum $\boldsymbol{x}_\alpha$ as perturbed from its true value $\bar{\boldsymbol{x}}_\alpha$ by $\Delta\boldsymbol{x}_\alpha$, which is assumed to be independent Gaussian noise of mean $\boldsymbol{0}$ and covariance matrix $V[\boldsymbol{x}_\alpha]$. Then, maximum likelihood is equivalent to the minimization of the *Mahalanobis distance*

$$I = \sum_{\alpha=1}^{N} (\bar{\boldsymbol{x}}_\alpha - \boldsymbol{x}_\alpha, V[\boldsymbol{x}_\alpha]^{-1}(\bar{\boldsymbol{x}}_\alpha - \boldsymbol{x}_\alpha)), \qquad (1)$$

with respect to the true values $\bar{\boldsymbol{x}}_\alpha$ subject to given knowledge about them. Hereafter, $(\boldsymbol{a}, \boldsymbol{b})$ denotes the inner product of vectors $\boldsymbol{a}$ and $\boldsymbol{b}$.

If the noise is homogeneous and isotropic, in which case $V[\boldsymbol{x}_\alpha] = c\boldsymbol{I}$ for all α for some constant c and the unit matrix $\boldsymbol{I}$, the Mahalanobis distance I is equivalent to the sum of the squares of the geometric distances between the observations $\boldsymbol{x}_\alpha$ and their true values $\bar{\boldsymbol{x}}_\alpha$, often referred to as the *reprojection error*. That name originates from the following intuition: In inferring the 3-D structure of the scene from its projected images, maximum likelihood under homogeneous and isotropic Gaussian noise means *reprojecting* the inferred 3-D structure onto the images and minimizing the square distance between the *reprojection* of the solution and the projection of the scene. Reprojection error minimization is also referred to as *geometric fitting*.

Theory

The estimation procedure depends on the way the knowledge about true values $\bar{\boldsymbol{x}}_\alpha$ is represented. A typical approach is to introduce some function $\boldsymbol{g}(\boldsymbol{t}, \boldsymbol{\theta})$ to express $\bar{\boldsymbol{x}}_\alpha$ in a parametric form

$$\bar{\boldsymbol{x}}_\alpha = \boldsymbol{g}(\boldsymbol{t}_\alpha, \boldsymbol{\theta}), \qquad (2)$$

where $\boldsymbol{t}_\alpha$ is a control variable that specifies the identity of the αth datum and $\boldsymbol{\theta}$ is an unknown parameter

that specifies the underlying structure. After (2) is substituted, the Mahalanobis distance I becomes a function of $\boldsymbol{\theta}$ alone, which is then minimized with respect to $\boldsymbol{\theta}$. This is the standard approach in the traditional statistic estimation framework and also known as *regression*.

This parametric approach, however, is quite limited in computer vision applications. Often, no such knowledge as (2) is available about the true values $\bar{\boldsymbol{x}}_\alpha$ except that they satisfy some implicit equations of the form

$$F^{(k)}(\boldsymbol{x}, \boldsymbol{\theta}) = 0, \qquad k = 1, \ldots, L. \tag{3}$$

The unknown parameter $\boldsymbol{\theta}$ allows one to infer the 2-D/3-D shape and motion of the objects observed in the images.

This type of estimation leads to some theoretical problems. Usually, no restriction is imposed on the true values $\bar{\boldsymbol{x}}_\alpha$ except that they should satisfy (3). This is called the *functional model*. One could alternatively introduce some statistical model according to which the true values $\bar{\boldsymbol{x}}_\alpha$ are sampled. Then, the model is called *structural*. This distinction is crucial when one considers limiting processes in the following sense. Traditional statistical analysis mainly focuses on the asymptotic behavior as the number of observations increases to ∞. This is based on the reasoning that the mechanism underlying noisy observations would better reveal itself as the number of observations increases (the law of large numbers) while the number of available data is limited in practice. So, the estimation accuracy vs. the number of data is a major concern. In this light, efforts have been made to obtain a consistent estimator in the sense that the solution approaches its true value in the limit $N \to \infty$ of the number N of the data.

In computer vision applications, in contrast, one cannot *repeat* observations. One makes an inference that given a single set of images and how many times one applies image processing operations, the result is always the same because standard image processing algorithms are deterministic and no randomness is involved. This is in a stark contrast to conventional statistical problems where observations are viewed as *samples* from potentially infinitely many possibilities and could obtain, by repeating observations, different values originating from unknown, uncontrollable, or unmodeled causes, which is called *noise* as a whole.

In vision problems, the accuracy of inference deteriorates as the uncertainty of image processing operations increases. Thus, the inference accuracy vs. the uncertainty of image operations, which is called *noise* for simplicity, is a major concern. Usually, the noise is very small, often subpixel levels. In light of this observation, it has been pointed out that in image domains the *consistency* of estimators should more appropriately be defined by the behavior in the limit $\sigma \to 0$ of the noise level σ [1, 3]. The functional model suits this purpose. If the error behavior in the limit of $N \to \infty$ were to be analyzed, one needs to assume some structural model that specifies how the statistical characteristics of the data depend on N. However, it is difficult to predict the noise characteristics for different N. Image processing filters usually output a list of points or lines or their correspondences along with their confidence values, from which only those with high confidence are used. If a lot of data are to be collected, those with low confidence need to be included, but their statistical properties are hard to estimate, since such data are possibly misdetections. This is the most different aspect of image processing from laboratory experiments, in which any number of data can be collected by repeated trials.

Maximum Likelihood with Implicit Constraints

Maximum likelihood based on the functional model is to minimize (1) subject to implicit constraints in the form of (3). In statistics, maximum likelihood is criticized for its lack of consistency. In fact, estimation of the true values $\bar{\boldsymbol{x}}_\alpha$, called *nuisance parameters* when viewed as parameters, is not consistent as $N \to \infty$ in the maximum likelihood framework [6]. However, the lack of consistency has no realistic meaning in vision applications as explained above. On the contrary, maximum likelihood has very desirable properties in the limit $\sigma \to 0$ of the noise level σ: The solution is *consistent* in the sense that it converges to the true value as $\sigma \to 0$ and *efficient* in the sense that its covariance matrix approaches a theoretical lower bound as $\sigma \to 0$ [1, 3].

According to the experience of many vision researchers, maximum likelihood is known to produce highly accurate solutions, and no necessity is felt for

further accuracy improvement. Rather, a major concern is its computational burden because maximum likelihood usually requires complicated nonlinear optimization. The standard approach is to introduce some auxiliary parameters to express each of $\bar{\boldsymbol{x}}_\alpha$ explicitly in terms of $\boldsymbol{\theta}$ and the auxiliary parameters. After they are substituted back into (1), the Mahalanobis distance I becomes a function of $\boldsymbol{\theta}$ and the auxiliary parameters. Then, this joint parameter space, which usually has very high dimensions, is searched for the minimum. This approach is called *bundle adjustment*, a term originally used by photogrammetrists. This is very time consuming, in particular if one seeks a globally optimal solution by searching the entire parameter space exhaustively.

Linear Reparameterization

In many important vision applications, the problem can be reparameterized to make the functions $F^{(k)}(\boldsymbol{x}, \boldsymbol{\theta})$ linear in $\boldsymbol{\theta}$ (but generally nonlinear in $\boldsymbol{x}$), allowing one to write (3) as

$$(\boldsymbol{\xi}^{(k)}(\boldsymbol{x}), \boldsymbol{\theta}) = 0, \qquad k = 1, \ldots, L, \tag{4}$$

where $\boldsymbol{\xi}^{(k)}(\boldsymbol{x})$ represents a nonlinear mapping of $\boldsymbol{x}$. This formalism covers many fundamental problems of computer vision including fitting a parametric curve such as a line, an ellipse, and a polynomial curve to a noisy 2-D point sequence or a parametric surface such as a plane, an ellipsoid, and a polynomial surface to a noisy 3-D point sets and computing the fundamental matrix or the homography from noisy point correspondences over two images. For this type of problem, a popular alternative to bundle adjustment is minimization of a function of $\boldsymbol{\theta}$ alone, called the *Sampson error*. Let us abbreviate $\boldsymbol{\xi}^{(k)}(\boldsymbol{x}_\alpha)$ to $\boldsymbol{\xi}_\alpha^{(k)}$. The first-order variation of $\boldsymbol{\xi}_\alpha^{(k)}$ by noise is

$$\Delta\boldsymbol{\xi}_\alpha^{(k)} = \boldsymbol{T}_\alpha^{(k)} \Delta\boldsymbol{x}_\alpha, \quad \boldsymbol{T}_\alpha^{(k)} \equiv \left. \frac{\partial \boldsymbol{\xi}^{(k)}(\boldsymbol{x})}{\partial \boldsymbol{x}} \right|_{\boldsymbol{x}=\bar{\boldsymbol{x}}_\alpha}. \tag{5}$$

Define the covariance matrices of $\boldsymbol{\xi}_\alpha^{(k)}$, $k = 1, \ldots, L$, by

$$\begin{aligned} V^{(kl)}[\boldsymbol{\xi}_\alpha] &= E[\Delta\boldsymbol{\xi}_\alpha^{(k)} \Delta\boldsymbol{\xi}_\alpha^{(l)\top}] \\ &= \boldsymbol{T}_\alpha^{(k)} E[\Delta\boldsymbol{x}_\alpha \Delta\boldsymbol{x}_\alpha^\top] \boldsymbol{T}_\alpha^{(l)\top} \\ &= \boldsymbol{T}_\alpha^{(k)} V[\boldsymbol{x}_\alpha] \boldsymbol{T}_\alpha^{(l)\top}, \end{aligned} \tag{6}$$

where $E[\cdot]$ denotes expectation. The Sampson error that approximates the minimum of the Mahalanobis distance I subject to the constraints in (4) has the form

$$K = \sum_{\alpha=1}^{N} \sum_{k,l,=1}^{L} W_\alpha^{(kl)}(\boldsymbol{\xi}_\alpha^{(k)}, \boldsymbol{\theta})(\boldsymbol{\xi}_\alpha^{(l)}, \boldsymbol{\theta}), \tag{7}$$

where $W_\alpha^{(kl)}$ is the (kl) element of $(\boldsymbol{V}_\alpha)_r^-$. Here, $\boldsymbol{V}_\alpha$ is the matrix whose (kl) element is

$$\boldsymbol{V}_\alpha = \Big((\boldsymbol{\theta}, V^{(kl)}[\boldsymbol{\xi}_\alpha]\boldsymbol{\theta}) \Big), \tag{8}$$

where the true data values $\bar{\boldsymbol{x}}_\alpha$ in the definition of $V^{(kl)}[\boldsymbol{\xi}_\alpha]$ are replaced by their observations $\boldsymbol{x}_\alpha$. The operation $(\cdot)_r^-$ denotes the pseudoinverse of truncated rank r, (i.e., with all eigenvalues except the largest r replaced by 0 in the spectral decomposition), and r is the rank (the number of independent equations) of (4). The name Sampson error stems from the classical ellipse fitting scheme [8].

The Sampson error (7) can be minimized by various means including the *FNS* (*fundamental numerical scheme*) [2], and the *HEIV* (*heteroscedastic errors-in-variable*) [5]. It can be shown that the exact maximum likelihood solution can be obtained by repeating Sampson error minimization, each time modifying the Sampson error so that in the end the modified Sampson error coincides with the Mahalanobis distance [4]. It turns out that in many practical applications the solution that minimizes the Sampson error coincides with the exact maximum likelihood solution up to several significant digits; usually, two or three rounds of Sampson error modification are sufficient.

It can be shown that the covariance matrix $V[\hat{\boldsymbol{\theta}}]$ of any unbiased estimator $\hat{\boldsymbol{\theta}}$ of $\boldsymbol{\theta}$ satisfies under some general conditions the inequality

$$V[\hat{\boldsymbol{\theta}}] \succ \left(\sum_{\alpha=1}^{N} \sum_{k,l=1}^{L} \bar{W}_\alpha^{(kl)} \bar{\boldsymbol{\xi}}_\alpha^{(k)} \bar{\boldsymbol{\xi}}_\alpha^{(l)\top} \right)_r^-, \tag{9}$$

where $\bar{\boldsymbol{\xi}}_\alpha^{(k)}$ are the true value of $\boldsymbol{\xi}_\alpha^{(k)}$ and $\bar{W}_\alpha^{(kl)}$ is the value of $W_\alpha^{(kl)}$ defined earlier evaluated for the

true values of $\boldsymbol{\xi}_\alpha^{(k)}$ and $\boldsymbol{\theta}$. The symbol $\succ$ means that the left-hand side minus the right-hand side is positive semidefinite. The right-hand side of (9) is called the *KCR* (*Kanatani-Cramer-Rao*) *lower bound* [1, 3]. It can be shown that the covariance matrix of Sampson error minimization solution coincides with this bound in the leading order in the noise level [1, 3].

Algebraic Methods

Sampson error minimization schemes such as FNS and HEIV rely on local search, and the iterations do not always converge unless started from a value sufficiently close to the solution. For accurate initialization of the iterations, various types of algebraic method have been studied. *Algebraic methods* refer to minimizing the *algebraic distance*

$$\begin{aligned} J &= \sum_{\alpha=1}^{N}\sum_{k=1}^{L}(\boldsymbol{\xi}_\alpha^{(k)},\boldsymbol{\theta})^2 \\ &= \sum_{\alpha=1}^{N}\sum_{k=1}^{L}\boldsymbol{\theta}^\top\boldsymbol{\xi}_\alpha^{(k)}\boldsymbol{\xi}_\alpha^{(k)\top}\boldsymbol{\theta} \\ &= (\boldsymbol{\theta},\boldsymbol{M}\boldsymbol{\theta}), \end{aligned} \tag{10}$$

where the matrix $\boldsymbol{M}$ is defined by

$$\boldsymbol{M} = \sum_{\alpha=1}^{N}\sum_{k=1}^{L}\boldsymbol{\xi}_\alpha^{(k)}\boldsymbol{\xi}_\alpha^{(k)\top}. \tag{11}$$

Note that if $W_\alpha^{(kl)}$ in (7) is replaced by the Kronecker delta δ_{kl}, the Sampson error K coincides with the algebraic distance J. Algebraic distance minimization is also referred to as *algebraic fitting* or simply *least squares*.

The algebraic distance J is trivially minimized by $\boldsymbol{\theta} = \boldsymbol{0}$ unless some scale normalization is imposed on $\boldsymbol{\theta}$. The standard normalization is $\|\boldsymbol{\theta}\| = 1$. A more general class of normalization is the form of $(\boldsymbol{\theta}, \boldsymbol{N}\boldsymbol{\theta}) = 1$, where $\boldsymbol{N}$ is some positive definite or semidefinite symmetric matrix. It is easily seen that the solution is given by solving the generalized eigenvalue problem

$$\boldsymbol{M}\boldsymbol{\theta} = \lambda\boldsymbol{N}\boldsymbol{\theta}, \tag{12}$$

for the smallest generalized eigenvalue λ. The choice of $\boldsymbol{N} = \boldsymbol{I}$ corresponds to the standard normalization. However, the solution depends on $\boldsymbol{N}$, and it has been reported by many researchers that the choice $\boldsymbol{N} = \boldsymbol{I}$ leads to large statistical bias. For example, the ellipse thus fitted to a point sequence is almost always smaller than the true one.

One naturally asks: What $\boldsymbol{N}$ will maximize the accuracy of the solution? It is widely recognized that the scheme due to Taubin [9] produces a fairly accurate estimate. Recently, it has been found that if $\boldsymbol{N}$ is allowed to be nondefinite, i.e., neither positive or negative (semi)definite, the statistical bias can be eliminated up to second-order noise terms, resulting in a method called *HyperLS* with slightly better performance than the Taubin method [7].

References

1. Chernov N, Lesort C (2004) Statistical efficiency of curve fitting algorithms. Comput Stat Data Anal 47(4):713–728
2. Chojnacki W, Brooks MJ, van den Hengel A, Gawley D (2000) On the fitting of surfaces to data with covariances. IEEE Trans Pattern Anal Mach Intell 22(11): 1294–1303
3. Kanatani K (2008) Statistical optimization for geometric fitting: theoretical accuracy analysis and high order error analysis. Int J Comput Vis 80(2):167–188
4. Kanatani K, Sugaya Y (2010) Unified computation of strict maximum likelihood for geometric fitting. J Math Imaging Vis 38(1):1–13
5. Leedan Y, Meer P (2000) Heteroscedastic regression in computer vision: Problems with bilinear constraint. Int J Comput Vis 37(2):127–150
6. Neyman J, Scott EL (1948) Consistent estimates based on partially consistent observations. Econometrica 16(1): 1–32
7. Rangarajan P, Kanatani K, Niitsuma H, Sugaya Y (2010) Hyper least squares and its applications. In: Proceedings of the 20th international conference on pattern recognition, August 2010, Istanbul, Turkey
8. Sampson PD (1982) Fitting conic sections to "very scattered" data: An iterative refinement of the Bookstein algorithm. Comput Graph Image Process 18(1): 97–108
9. Taubin G (1991) Estimation of planar curves, surfaces, and non-planar space curves defined by implicit equations with applications to edge and range image segmentation. IEEE Trans Pattern Anal Mach Intell 13(11):1115–1138

Optimal Parameter Estimation

▸Optimal Estimation

Oren-Nayar Reflectance Model

Ping Tan
Department of Electrical and Computer Engineering, National University of Singapore, Singapore, Singapore

Related Concepts

►Radiance; ►Reflectance Models

Definition

The Oren-Nayar reflectance model calculates the amount of reflected radiance at a surface point according to the lighting, viewing, and surface normal directions at that point. It is characterized by its high accuracy in modeling diffuse reflection for rough surfaces. It was introduced by Michael Oren and Shree K. Nayar [1] in 1994.

Background

When light arrives at surfaces, it can be reflected, refracted, scattered, or absorbed. Reflectance models are mathematical functions that describe the interactions between light and surfaces. Usually, they are functions of lighting, viewing, and surface normal directions. There are various reflectance models at different levels of precision and complexity. Most reflectance models include components describing diffuse and specular reflection, respectively.

The Oren-Nayar model is a reflectance model that accurately represents the diffuse reflection of rough surfaces. In the field of computer vision, diffuse reflection is often modeled by the Lambert's model [2] which assumes the light is uniformly reflected in all reflected directions. However, many real surfaces, especially rough surfaces such as clay or concrete, exhibit significant non-Lambertian diffuse reflection. For example, the reflection is often stronger when the viewing direction is close to the lighting direction, which is called *backscattering*. Oren and Nayar designed a more accurate model to describe diffuse reflection of rough surfaces. This model can be applied to generate realistic computer graphic images or to accurately infer scene properties such as shape and material from observed image intensities.

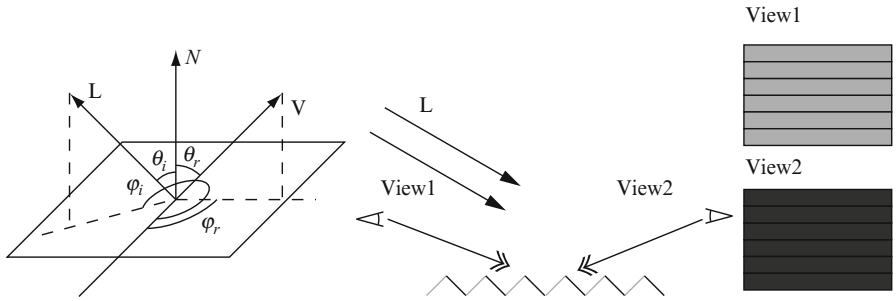
Theory

Light reflected by a surface can be roughly divided into specular and diffuse reflection. The specular reflection refers to the light reflected directly at the surface without entering it. Specular reflection typically is concentrated within a small angle in space. In comparison, diffuse reflection refers to the light that enters the surface and is distributed more uniformly in all reflected directions.

In computer vision, the Lambert's reflectance model is often used to describe diffuse reflection, which assumes that the diffuse reflection is equally distributed on a hemisphere indicating all the reflected directions. However, real surfaces, especially rough surfaces like clay and concrete, often exhibit non-Lambertian reflection. A common non-Lambertian phenomenon is the *backscattering* where more light is reflected toward the incident direction and the surface appears brighter when the viewing direction is closer to the lighting direction. For example, this phenomenon is observed in lunar photometry and is modeled by Opik [3] and Minnaert [4].

Backscattering can be well explained by studying the surface microstructures. As shown on the right of Fig. 1, when the polar angle of the incident light θ_i is large, some of the microstructures are in shadows. When the surface is viewed from the direction of the incident lighting L, all visible microstructures are illuminated and the surface appears brighter. However, when the surface is viewed from the direction V, all visible microstructures are shadowed and the surface appears darker.

Oren and Nayar adopted a microfacet surface model to derive the diffuse reflection of rough surfaces, where surfaces consist of many infinitely long V-cavities and each cavity facet is Lambertian. The reflectance model of a surface is derived by integrating the reflection of all facets. A similar microfacet model is used by the Cook-Torrance reflectance model [5] to study specular reflection, where each microfacet acts as a mirror. The Oren-Nayar model is derived in three steps in the original paper [1]. First, a reflectance model $I^{(1)}$ is derived for anisotropic surfaces consisting of V-cavities of



Oren-Nayar Reflectance Model, Fig. 1 *Left*: directions involved in the definition of the Oren-Nayar model. N is the surface normal direction, and L and V are the lighting and viewing direction, respectively. *Right*: a rough surface consists of many microfacets. When the polar angle of the incident light θ_i is large, some of the microfacets are in shadow and some of them are illuminated, as illustrated by darker and brighter segments. The surface appears brighter in the view 1 than in the view 2, because most of the visible microfacets in view 1 are illuminated and those in view 2 are shadowed

the same slop and the same direction. This model includes two components, one for direction illumination and the other for interreflection between microfacets. Second, a reflectance model $I^{(2)}$ is derived for isotropic surfaces consisting of V-cavities with the same slop but uniformly distributed directions by integrating $I^{(1)}$ over different cavity directions. Third, the final Oren-Nayar model is derived for general isotropic surfaces that consist of V-cavities whose slops follow a Gaussian distribution with mean μ and variance σ^2 and whose directions follow a uniform distribution. The Oren-Nayar reflectance model $I^{(3)}$ can be derived by integrating $I^{(2)}$ over different cavity slopes. This model also includes a component for direction illumination

$$I_1^{(3)}(\theta_r,\theta_i,\phi_r-\phi_i;\sigma)=\frac{\rho}{\pi}E_0\cos\theta_i\Bigg[C_1(\sigma)+\cos(\phi_r-\phi_i)C_2(\alpha;\beta;\phi_r-\phi_i;\sigma)\tan\beta+(1-|\cos(\phi_r-\phi_i)|)C_3(\alpha;\beta;\sigma)\tan\left(\frac{\alpha+\beta}{2}\right)\Bigg] \quad (1)$$

and a component for microfacet interreflection

$$I_2^{(3)}(\theta_r,\theta_i,\phi_r-\phi_i;\sigma)=0.17\frac{\rho^2}{\pi}E_0\cos\theta_i\frac{\sigma^2}{\sigma^2+0.13}\left[1-\cos(\phi_r-\phi_i)\left(\frac{2\beta}{\pi}\right)^2\right]. \quad (2)$$

Here, (θ_i,ϕ_i) and (θ_r,ϕ_r) are the polar and azimuth angles of the lighting and viewing directions, respectively, as illustrated on the left of Fig. 1, $\alpha=\max(\theta_r,\theta_i)$, $\beta=\min(\theta_r,\theta_i)$, E_0 indicates the illumination intensity and ρ is the albedo in the Lambert's reflectance model of microfacet. C_1, C_2, and C_3 are evaluated according to the following equations:

$$C_1=1-0.5\frac{\sigma^2}{\sigma^2+0.33}$$

$$C_2=\begin{cases}0.45\frac{\sigma^2}{\sigma^2+0.09}\sin\alpha & \text{if } \cos(\phi_r-\phi_i)\geq 0\\ 0.45\frac{\sigma^2}{\sigma^2+0.09}\left(\sin\alpha-\left(\frac{2\beta}{\pi}\right)^3\right) & \text{otherwise}\end{cases}$$

$$C_3=0.125\left(\frac{\sigma^2}{\sigma^2+0.09}\right)\left(\frac{4\alpha\beta}{\pi^2}\right)^2.$$

Experiments reported in [1] show that the Oren-Nayar reflectance model matches the measured reflectance data from rough surfaces well and represents the *backscattering*. An interesting fact about this model is that it degenerates to the Lambert's model when $\sigma^2=0$. In applications, to reduce computation and simplify analysis, a simplified version of the Oren-Nayar model is often desired. The term C_3 and the interreflection are found to be relatively small during simulation. Hence, a simplified model can be obtained by discarding C_3 and ignoring interreflection:

$$I(\theta_r,\theta_i,\phi_r-\phi_i;\sigma)=\frac{\rho}{\pi}E_0\cos\theta_i(A+B\max[0,\cos(\phi_r-\phi_i)]\sin\alpha\tan\beta) \quad (3)$$

Here, $A=1.0-0.5\frac{\sigma^2}{\sigma^2+0.33}$ and $B=0.45\frac{\sigma^2}{\sigma^2+0.09}$.

Application

The Oren-Nayar reflectance model computes reflected radiance according to the 3D shape, viewing, and lighting configurations. Like many other reflectance models, it can be used to create computer graphic images. It is shown in [1] that the Oren-Nayar reflectance model can generate images of rough surfaces more realistically than the Lambert's model. This reflectance model was also applied for photometric stereo in [6] and generated better results than the Lambertian photometric stereo algorithm.

References

1. Oren M, Nayar SK (1994) Generalization of lambert's reflectance model. In: SIGGRAPH '94: proceedings of the 21st annual conference on computer graphics and interactive techniques. ACM, New York, pp 239–246
2. Lambert JH (1760) Photometria sive de mensure de gratibus lumi-nis, colorum umbrae. Eberhard Klett
3. Opik E (1924) Photometric measures of the moon and the moon the earth-shine. Publications de L'Observatorie Astronomical de L'Universite de Tartu 26(1):1–68
4. Minnaert M (1941) The reciprocity principle in lunar photometry. Astrophys J 93:403–C410
5. Cook RL, Torrance KE (1981) A reflectance model for computer graphics. In: SIGGRAPH '81: proceedings of the 8th annual conference on computer graphics and interactive techniques. ACM, New York, pp 307–316
6. Oren M, Nayar SK (1995) Generalization of the lambertian model and implications for machine vision. Int J Comput Vis 14(3):227–251

Osculating Paraboloids

Jan J. Koenderink
Faculty of EEMSC, Delft University of Technology, Delft, The Netherlands
The Flemish Academic Centre for Science and the Arts (VLAC), Brussels, Belgium
Laboratory of Experimental Psychology, University of Leuven (K.U. Leuven), Leuven, Belgium

Synonyms

Osculating quadric; Second order approximation

Related Concepts

▸Curvature; ▸Differential Invariants

Definition

Osculating paraboloids commonly occur in second-order approximations where the first order is irrelevant or can be transformed away.

Background

Despite the common occurrence of osculating paraboloids in many settings, there appears to be no literature dedicated to their shape space. Although the standard taxonomy of quadrics in Euclidean space is familiar to most, what is missing is the geometrical structure of the manifold of all osculating paraboloids, inclusive a metric. Here the two-parameter case (important in many applications) is discussed in a little detail.

Theory

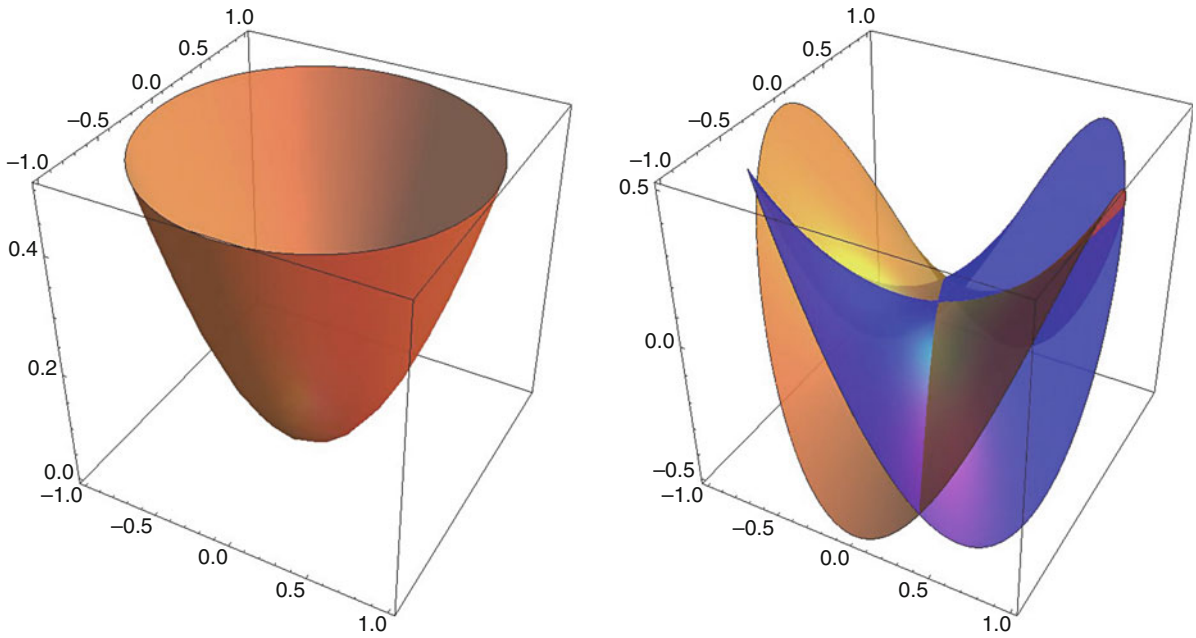
"Osculating paraboloids" are second-order Monge patches:

$$x\mathbf{e}_x + y\mathbf{e}_y + z(x,y)\mathbf{e}_z = x\mathbf{e}_x + y\mathbf{e}_y + \frac{1}{2}(a_{20}x^2 + 2a_{11}xy + a_{02}y^2)\mathbf{e}_z, \quad (1)$$

that occur in many contexts. Apparently they inhabit a three-dimensional quadric shape space that may be parameterized by the coefficient triple $\{a_{20}, a_{11}, a_{02}\}$.

Typically the z-domain will be some physical parameter with a physical dimension different from that of the xy-domain. The appropriate setting then is not Euclidean. In most applications one treats the z-dimension as isotropic, then the space is a "singly isotropic space," or "graph space."

Scaling and addition in this space correspond to point-wise addition of heights, thus is well defined and makes geometrical sense. The quadric shape space is a linear space under addition and multiplications with real numbers. Thus it is of some interest to find a basis that makes intuitive, and/or pragmatic sense.



Osculating Paraboloids, Fig. 1 The basis of osculating quadrics. At *left* the isotropic paraboloid $(x^2 + y^2)/2$, at *right* the anisotropic paraboloids xy and $(x^2 - y^2)/2$, these are mutually congruent

One lead is that all such surfaces that differ only by a rotation about the z-axis ($\mathbf{e}_z$) are geometrically congruent and may often be grouped as a single "shape." This leads one to consider the isotropic paraboloid:

$$\frac{x^2 + y^2}{2}, \tag{2}$$

to be "special," since it is invariant under such rotations. Moreover, the pair:

$$xy, \quad \frac{x^2 - y^2}{2}, \tag{3}$$

is likewise "special" because they are the same under a rotation over $\frac{\pi}{4}$ (notice that all quadrics transform into themselves under rotations over π). These shapes transform into their "negatives" under rotations of $\frac{\pi}{2}$, thus, in a sense, they are their own negatives. This is not the case in general, for instance, $x^2 + y^2$ cannot be transformed into its negative (i.e., $-(x^2 + y^2)$) under any rotation.

Thus the basis of quadrics contains only two distinct shapes (Fig. 1), rather than three as one might naively expect. Using this basis one evidently has:

$$\begin{aligned} &\frac{1}{2}(a_{20}x^2 + 2a_{11}xy + a_{02}y^2) \\ &= r\frac{x^2 - y^2}{2} + sxy + t\frac{x^2 + y^2}{2}, \end{aligned} \tag{4}$$

with:

$$r = \frac{a_{20} - a_{02}}{2}, \quad s = a_{11}, \quad t = \frac{a_{20} + a_{02}}{2}. \tag{5}$$

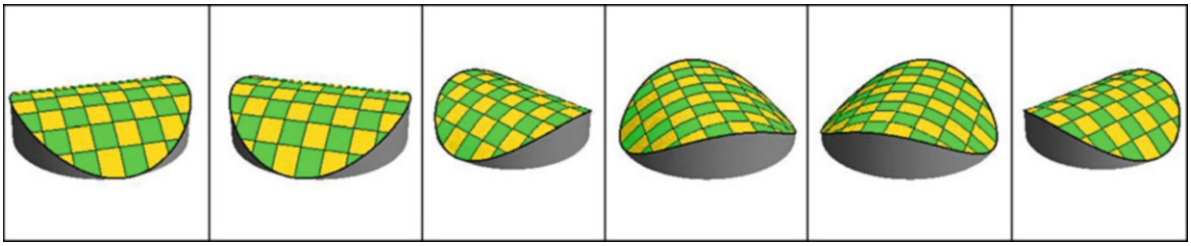
By a suitable rotation any quadric may be written in the canonical form

$$\frac{1}{2}(\kappa_1 u^2 + \kappa_2 v^2), \text{ with } \kappa_1 \geq \kappa_2. \tag{6}$$

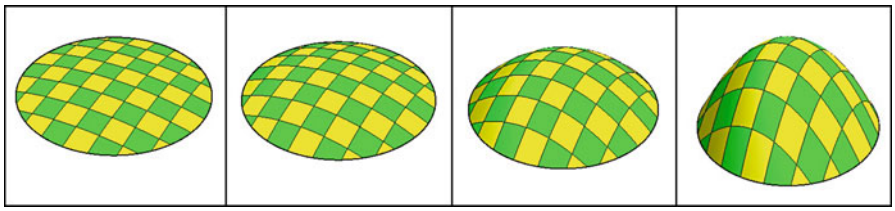
Such a canonical form is convenient because it abstracts away from the orientation about the z-axis (Fig. 2). The general quadric is rotated by angle φ with respect to the u-axis (the first principal direction), where $-\frac{\pi}{2} < \varphi < \frac{\pi}{2}$. One has:

$$a_{20} = \frac{\kappa_1 + \kappa_2}{2} + \frac{\kappa_1 - \kappa_2}{2}\cos 2\varphi, \tag{7}$$

$$a_{11} = \frac{\kappa_1 - \kappa_2}{2}\sin 2\varphi, \tag{8}$$



Osculating Paraboloids, Fig. 2 A suite of surfaces of the same Casorati curvature and shape parameter but different orientations. The orientation domain is periodic with period π



Osculating Paraboloids, Fig. 3 A scale of surfaces of the same shape, the Casorati curvature varying by factors of two

$$a_{02} = \frac{\kappa_1 + \kappa_2}{2} - \frac{\kappa_1 - \kappa_2}{2} \cos 2\varphi, \tag{9}$$

and, equivalently:

$$r = \frac{\kappa_1 - \kappa_2}{2} \cos 2\varphi, \tag{10}$$

$$s = \frac{\kappa_1 - \kappa_2}{2} \sin 2\varphi, \tag{11}$$

$$t = \frac{\kappa_1 + \kappa_2}{2}. \tag{12}$$

Introduce the Casorati curvature (see Fig. 3) as:

$$\kappa = \sqrt{\frac{\kappa_1^2 + \kappa_2^2}{2}}, \tag{13}$$

and the shape parameter (see Fig. 4) as:

$$\sigma = \arctan \frac{\kappa_1 + \kappa_2}{\kappa_1 - \kappa_2}. \tag{14}$$

These differential invariants can be written into various forms, e.g.,

$$\kappa = \sqrt{\frac{a_{20}^2 + 2a_{11}^2 + a_{02}^2}{2}} = \sqrt{r^2 + s^2 + t^2}, \tag{15}$$

and

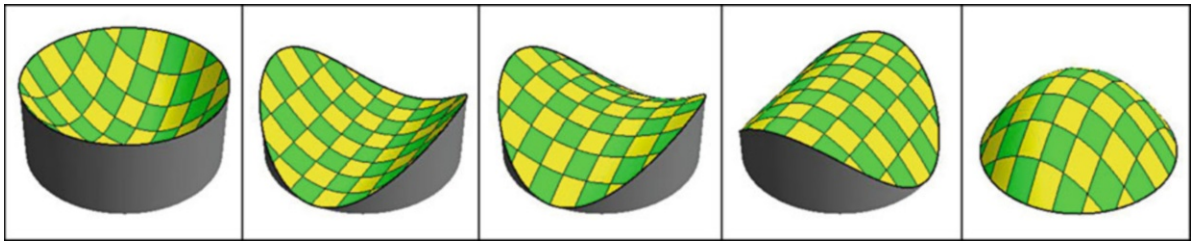
$$\tan \sigma = \frac{a_{20} + a_{02}}{\sqrt{(a_{20} - a_{02})^2 + 4a_{11}^2}} = \frac{t}{\sqrt{r^2 + s^2}}. \tag{16}$$

This can be further simplified by identifying the mean curvature $2H = \kappa_1 + \kappa_2 = a_{20} + a_{02}$, the Gaussian curvature $K = \kappa_1\kappa_2 = a_{20}a_{02} - a_{11}^2$, and the "bending energy" $E = \kappa_1^2 + \kappa_2^2 = a_{20}^2 + 2a_{11}^2 + a_{02}^2$. (Notice that H and K should be distinguished from the invariants of the same name in Euclidean differential geometry. They correspond to the case of infinitesimal height, or to isotropic geometry.) The expression $\frac{1}{2}\sqrt{r^2 + s^2}$ captures the anisotropy of the quadric and may well be denoted its "non-umbilicity."

The Casorati curvature is perhaps less well known, and it may be motivated as follows. One has:

$$\kappa = \sqrt{\frac{\kappa_1^2 + \kappa_2^2}{2}}, \tag{17}$$

and thus κ vanishes only for the planar case. This is different for the mean curvature $2H = \kappa_1 + \kappa_2$, which vanishes for minimal surfaces ($t = 0$), or the Gaussian curvature $K = \kappa_1\kappa_2$, which vanishes for cylindrical surfaces. This is often confusing to the beginner, for whom minimal surfaces and cylinders are evidently "curved." Moreover, one easily verifies that when the "spatial average" and "standard deviation" of a function $F(x, y)$ are defined as:



Osculating Paraboloids, Fig. 4 A suite of surfaces of the same Casorati curvature, the shape parameter taking values of $+\pi/2$, $+\pi/4$, 0, $-\pi/4$, and $-\pi/2$

$$\langle F(x,y)\rangle_\mu = \int_{\mathbb{R}^2} F(x,y)\left(\frac{\mathrm{e}^{-\frac{x^2+y^2}{2\mu^2}}}{2\pi\mu^2}\right)\mathrm{d}x\,\mathrm{d}y, \quad (18)$$

$$\text{and } [F]_\mu = \sqrt{\langle F^2\rangle_\mu - \langle F\rangle_\mu^2},$$

one has:

$$\kappa = \frac{\left[\kappa_1 x^2 + \kappa_2 y^2\right]}{[x^2+y^2]} \quad (19)$$

(where $\mu > 0$ has fallen out of the equation). Thus the Casorati curvature is essentially *the root mean square deviation from planarity*.

The shape parameter σ is best understood as a measure of the ratio of umbilicity (isotropy) to non-umbilicity (anisotropy) of the shape. Shapes that differ only in the sign of the shape parameters stand in the relation as a cast to its mold. The minimal shapes ($\sigma = 0$) "are their own molds" as they can be fitted into their negatives after a rotation over $\frac{\pi}{2}$.

Now consider polar coordinates in shape space, defined as:

$$\varrho = \sqrt{r^2+s^2+t^2} = \kappa \quad (20)$$

$$\vartheta = \arctan\frac{t}{\sqrt{r^2+s^2}} = \sigma, \quad (21)$$

$$\phi = \arctan\frac{s}{r} = 2\varphi. \quad (22)$$

The spheres concentric with the origin are loci of constant Casorati curvature, and the origin represents the planar case. A "shape" is a right circular cone with the t-axis as its axis. The t-axis itself is such a (degenerated) cone, it represents the umbilics. The rs-plane is also such a (again, degenerated) cone, it represents the minimal surfaces, which are the surfaces of zero mean curvature. The cone with semi-top angle of $\frac{\pi}{2}$ is the locus of parabolic (cylindrical) surfaces. Each half-plane on the t-axis houses the complete zoo of shapes up to orientation. A meridian contains all shapes of the same curvature, a latitude circle a single shape in all orientations. Thus one obtains a complete overview of all quadric shapes in a very natural parameterization (Figs. 5–8).

On a general curved surface the shape of the local osculating paraboloids will change from point to point (see Figs. 5–7). Since the change will be smooth, the surface can be mapped on a surface in shape space, a surface that may well be expected to have self-intersections and perhaps not everywhere smooth (e.g., have edges of regression or swallowtails), but will be continuous. This allows one to draw a number of useful conclusions concerning generic surfaces, e.g.:

- Planar points do not occur.
- Umbilics are isolated points.
- As a surface is deformed umbilics come and go in pairs.
- Parabolic curves occur on curves; on a closed surface they are closed curves.
- Minimal points occur on curves in hyperbolic areas; on a closed surface they are closed curves.
- Convex an concave regions are mutually isolated through hyperbolic areas.

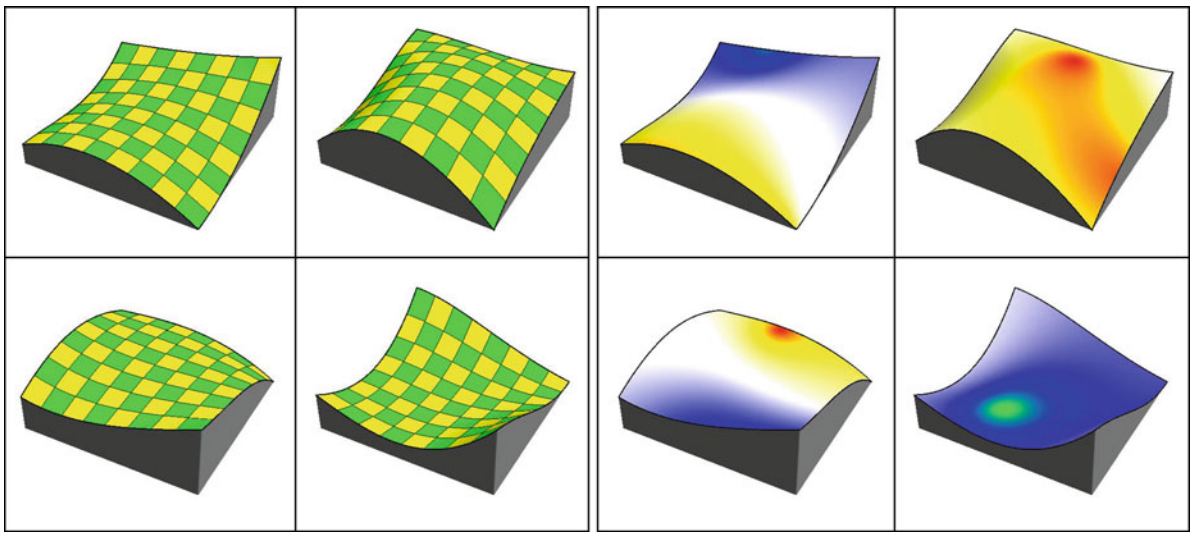
The local structure of these surfaces naturally involves the osculating cubics, which can be written as follows:

$$C(x,y) = \frac{1}{2!}(a_{20}x^2 + 2a_{11}xy + a_{02}y^2) + \frac{1}{3!}(a_{30}x^3 + 3a_{21}x^2y + 3a_{12}xy^2 + a_{03}y^3). \quad (23)$$

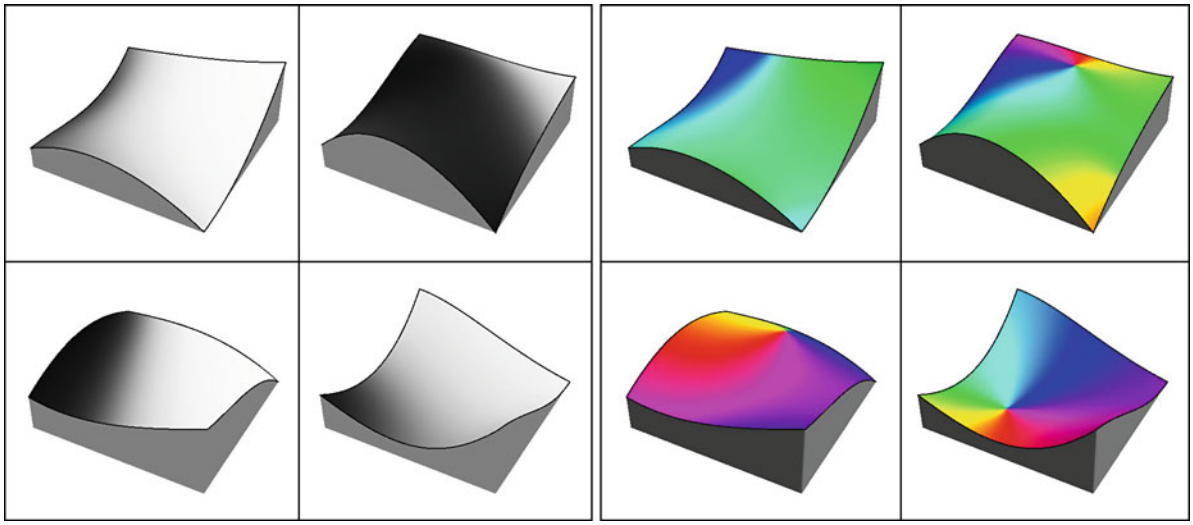
In an infinitesimal neighborhood of the origin one finds:

$$a'_{20}(x,y) = a_{20} + a_{30}x + a_{21}y, \quad (24)$$

$$a'_{11}(x,y) = a_{11} + a_{21}x + a_{12}y, \quad (25)$$



Osculating Paraboloids, Fig. 5 At *left* plots of four random surfaces. At *right* the surfaces have been color coded with the shape parameter. The color scale uses Hering's "opposite colors" (G: *Gegenfarben*). The umbilics are *red* (convex) and *green* (concave), the parabolic points *yellow* (ridge) and *blue* (rut), whereas the minimal points (zero mean curvature, symmetric saddles) are *white*



Osculating Paraboloids, Fig. 6 At *left* the surfaces of Fig. 5 left have been shaded with the Casorati curvature. The *gray scale* represents a nonlinear monotonic function of the logarithm of the curvature. At right the surfaces have been colored with the orientation of the direction of largest principal curvature. Notice the singularities at umbilical points

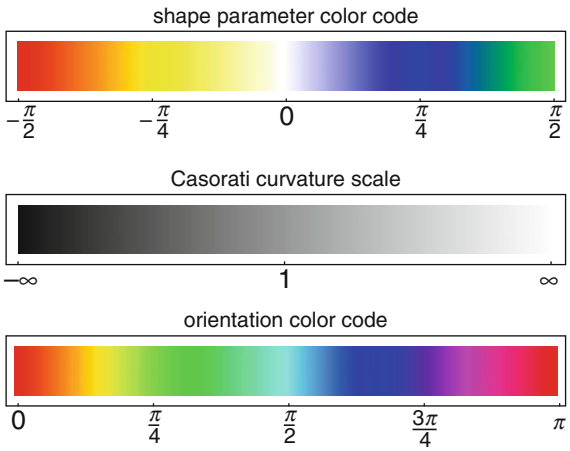
$$a'_{02}(x, y) = a_{02} + a_{12}x + a_{03}y. \tag{26}$$

Thus the quadric is perturbed by a linear combination of the two vectors in $\{r, s, t\}$-space:

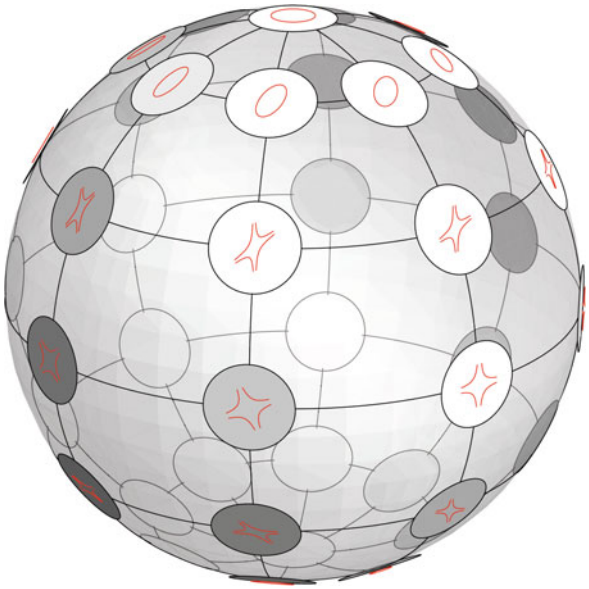
$$\frac{\partial}{\partial x}\{r, s, t\} = \frac{1}{2}\{a_{30} - a_{12}, 2a_{21}, a_{30} + a_{12}\} \tag{27}$$

$$\frac{\partial}{\partial y}\{r, s, t\} = \frac{1}{2}\{a_{21} - a_{30}, 2a_{12}, a_{03} + a_{21}\} \tag{28}$$

with weights x and y. These tangent vectors span a planar element in $\{r, s, t\}$-space. The planar element will in general be nondegenerate, the condition being that the two tangent vectors are independent, implying the



Osculating Paraboloids, Fig. 7 Scales used to indicate the parameters of osculating quadrics; these scales are used in Figs. 5 and 6



Osculating Paraboloids, Fig. 8 A sphere of constant Casorati curvature. The shapes have been indicated by their Dupin indicatrices. The equator has all orientations of the symmetric saddle, and each meridian has all shape indices with the umbilics at the poles. Each latitude circle repeats a shape at all orientations

cubic indicatrix of Dupin to have a single branch. Thus a neighborhood of a generic surface point maps on a surface in shape space. This allows one to infer various generic properties in a most simple manner, for instance that generic umbilics will be isolated points, the locus of parabolic points will be a curve on the surface, and so forth.

The osculating quadrics shape space may also be used to measure shape differences, a likely metric being:

$$d\ell^2 = \frac{dr^2 + ds^2 + dt^2}{r^2 + s^2 + t^2}, \qquad (29)$$

which is invariant with respect to rotations and homotheties about the origin. The metric can be rewritten as:

$$d\ell^2 = (d\log\kappa)^2 + d\sigma^2 + \sin^2\sigma\, d\phi^2, \qquad (30)$$

from which one sees that the geodesics are planar logarithmic spirals in planes through the origin. Lines through the origin and circles with the center at the origin are degenerated cases. For shapes of the same Casorati curvature the distance is simply a spherical arc length; in case the shapes have the same orientation this becomes the shape parameter, and for minimal surfaces it is the orientation difference. For shapes of the same shape parameter and orientation the distance is the logarithm of the ratio of Casorati curvatures, thus independent of the unit of length (or absolute size).

The osculating paraboloids shape space has numerous applications in very diverse contexts.

Open problem

The shape space concept described here appears to be little known and there is a decided lack of useful literature. The Casorati curvature and shape parameter appear in the literature as "curvedness" and (in a scaled version) as "shape index."

References

1. Griffin LD (2007) The 2nd order local-image–structure solid. IEEE Trans Pattern Anal Mach Intell 29(8):1355–1366
2. Koenderink JJ (1990) Solid shape. MIT, Cambridge, MA
3. Koenderink JJ, van Doorn AJ (1992) Surface shape and curvature scales. Image Vis Comput 10(8):557–564

Osculating Quadric

▸Osculating Paraboloids

Out of Focus Blur

▸Defocus Blur