O

Object Detection

Yali Amit[1], Pedro Felzenszwalb[2], and Ross Girshick[3]
[1]Department of Computer Science, University of Chicago, Chicago, IL, USA
[2]School of Engineering, Brown University, Providence, RI, USA
[3]Facebook AI Research, Menlo Park, CA, USA

Related Concepts

▸ Object Recognition

Definition

Object detection involves detecting instances of objects from one or several classes in an image.

Background

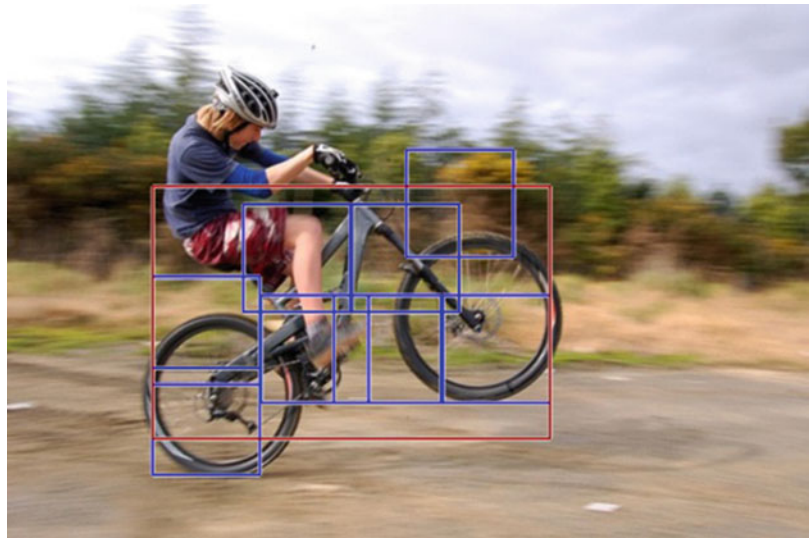
The goal of object detection is to detect all instances of objects from one or several known classes, such as people, cars, or faces in an image. Typically only a small number of objects are present in the image, but there is a very large number of possible locations and scales at which they can occur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

K. Ikeuchi (ed.), *Computer Vision*,
https://doi.org/10.1007/978-3-030-63416-2

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, a bounding box, or a segmentation mask. 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 (often hundreds or thousands) are necessary to capture certain aspects of class variability. Broadly speaking, less training data is needed when more information about class variability can be explicitly built into the model. However, it may be difficult to specify models that capture the vast variability found in images. An alternative approach is to use methods such as convolutional neural networks [1] that learn about class variability from large datasets.

Object detection approaches typically fall into one of two major categories, *generative* methods (see, e.g., [2–6]) and *discriminative* methods (see, e.g., [7–11]). A generative method consists of a probability model for the pose variability of the objects together with an appearance model:



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

a probability model for the image appearance conditional on a given pose, together with a model for background, i.e., non-object images. The model parameters can be estimated from training data, and the decisions are based on ratios of posterior probabilities. A discriminative method typically builds a classifier that can discriminate between images (or sub-images) containing instances of the target object classes and those not containing them. 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 7 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 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. In recent years, with the increase in annotated dataset size and computational acceleration using GPUs, this has been the approach of choice in the context of the multi-layer convolutional neural network paradigm, as discussed later in this article.

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 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 of 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 (see, e.g., [12]).

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 [10]. Local features are usually computed at a *dense* grid of locations in the image, leading to a dense feature map. Features such as HOG were designed by practitioners based on a variety of considerations involving the desired invariance and at times were motivated by certain analogies to biological processing in the visual system. An alternative approach, implemented in multi-layer convolutional neural networks, learns the local features as well as intermediate and higher-level features as part of the training process. Interestingly, the low-level features learned by such networks often resemble oriented edge detectors, like the designed features.

Local pooling of features is commonly used to introduce some degree of invariance to small geometric variations. A typical example is the *max* or *sum* operation [2, 13]. In this case a quantity that is to be computed at a pixel is replaced by the maximum or sum of the quantity in a neighborhood of the pixel. When the region is extended over the entire window, the result is a *bag of features* model [14], which counts 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 [15], corner detectors, and edge conjunctions [2]. 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 [8]. Another example are geometric edge arrangements of increasing complexity.

Latent variables An explicit parameterization of the object 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 [11]. 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

The most common approach to object detection reduces the problem to one of classification. Consider the problem of detecting instances from one object class 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 object detection problem to binary classification 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. Clearly the number of windows where the classifier needs to be computed can be prohibitive. Many computational approaches find ways to narrow down the number of windows where the classifier is implemented.

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 defines, the distribution of the image features in a window *conditional* on the pose, $p(X_{s+W}|object, \theta; \eta_a)$. The appearance model might be defined by 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} [2,3,5].

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 [4, 5]. 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," i.e., 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 [5]. Alternative background models involve subcollections of parts of the object model [16].

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 classifier to discriminate between image windows containing the target object classes and a broad background class. This is done using large amounts of data from the object classes and from background. 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* [8]. An initial classifier is trained to distinguish between the object and a manageable amount of background data. The classifier is designed to have very few 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.

Pose variables Certain discriminative models can also be implemented with latent pose parameters [11]. 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.

Convolutional neural networks Due to the limitations of low-level local features and the difficulty of manually specifying higher-level features, neural networks have become increasingly popular as a method to learn effective feature representations, from low-level to high-level, using large annotated datasets. These networks are composed of a hierarchy of layers indexed by grids of decreasing resolution. The input layer is the raw pixels of the input image. Each subsequent layer computes a vector output at each grid point using a list of local filters applied to the data in the preceding layer. This linear operation is typically followed by a nonlinear operation applied coordinate-wise and at certain layers the grid resolution is reduced by subsampling following a local max or averaging operation. The network terminates in one or more output layers that make predictions according to the design of the model (e.g., an object category

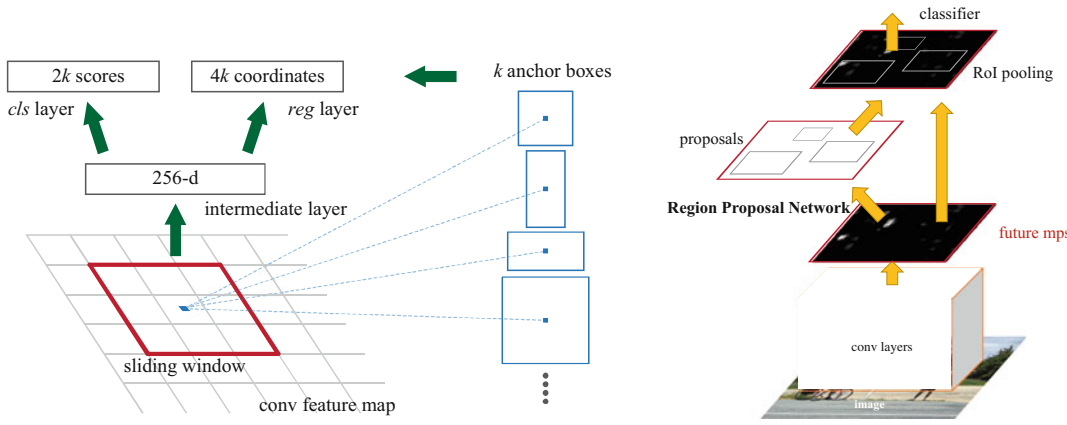
classifier and a pose estimator, if the model outputs pose information).

All of the filter coefficients and the parameters of the output layers are trainable. Training is done through stochastic gradient descent on a loss function defined in terms of the output layers and the labels in the dataset. Thus the network jointly learns linear classifiers and a complex hierarchy of nonlinear features that yield the final feature representation. Such networks were demonstrated to work for large-scale image classification tasks [17] and then subsequently for the more complex task of object detection [18].

Networks for image classification have a relatively straightforward design since they terminate with a single classification output for the entire image. Networks for object detection involve additional components that are designed to address the more complex nature of object detection. There are two dominant designs: *one stage* (e.g., [19]) and *two stage* (e.g., [20]), both of which are based on a sliding-window approach, described next.

In both of these designs, two convolutional sub-networks are applied in parallel. At each location on a relatively coarse feature grid, one of these sub-networks acts as a classifier and the other as a pose predictor (see Fig. 2 left "*cls* layer" and "*reg* layer," for classification and box regression, respectively). The pose estimator predicts a relative shift and scaling of a detection window, while the classifier predicts if it is an object or background. By creating multiple pairs of such layers, with each pair specializing to a window of a specific size and aspect ratio, the set of predefined sliding windows can better approximate the set of all possible image windows. The pose estimator is tasked with predicting the residual error between the quantized windows and the ground-truth object bounding boxes. In the first design paradigm, often termed "one-stage" methods, the sliding window classifier makes a multi-class prediction over the set of all object categories and background. These predictions, together with the refined windows, comprise the output of the model.

In the second design paradigm, the sliding-window classifier performs two-class classification between object (of any category) *vs.* background. The refined windows are then used as candidate object locations, often called regions of interest (RoIs), for a subsequent classification and window refinement stage. This second stage, as is typical in cascaded processing, only receives high-scoring RoIs from the object *vs.* background classifier. The input features to the



Object Detection, Fig. 2 Left: A sliding-window Region Proposal Network (RPN) that performs classification ("*cls* layer") and window shift and scaling regression ("*reg* layer") for a set of reference windows ("anchor boxes") at each grid position. Right: The Faster R-CNN system that uses RPN to generate candidate object proposals for processing in a second stage as a part of an overall convolutional neural network for object detection. (The figure is reproduced with permission from [20])

second stage are typically computed by extracting features within each RoI from a feature map. The RoI feature extraction process may involve quantizing the RoI coordinates and using max pooling [20] or bilinear interpolation of the feature map without performing quantization [21]. The output of the second stage is a multi-class classification prediction and window refinement (shift and scaling) for each RoI. See Fig. 2 right. Two-stage methods can also be extended in a straightforward manner to predict a binary segmentation mask for each RoI [21]. In either case, all parameters of these networks are trained jointly using stochastic gradient descent on a multi-task loss function that includes terms for classification and pose estimation.

Computational Methods

The basic detection process consists of searching over pose parameters to classify each hypothesis. At a minimum this usually involves searching over locations and sizes and is clearly a very intensive computation. There are a number of methods to make it more efficient.

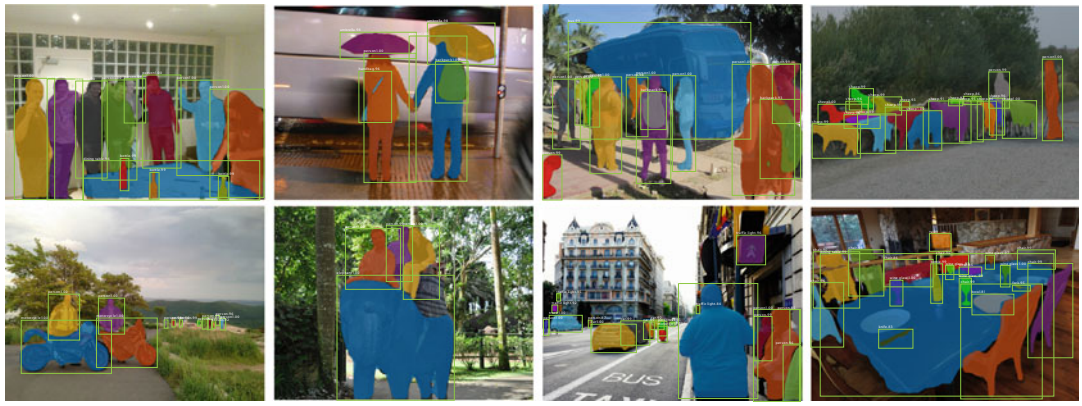
Sparse features When sparse features are used, it is possible to focus the computation only in regions around features. The two main approaches that take advantage of this sparsity are *alignment* [22] and the *generalized Hough transform*. Alignment uses information regarding the relative locations 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 [2].

Cascades As mentioned above the cascade method trains a sequence of classifiers with successively more difficult background data. Each such classifier is usually designed to be simple and 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 [9]. 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 [14] 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 [2, 3].

Convolutional neural networks Object detection systems based on convolutional networks make use of many classical techniques for reducing computation, including cascades as previously described. Coarse-to-fine approaches are also common. By progressively reducing spatial resolution, these networks operate on a coarse grid of object locations. The loss in resolution is compensated by predicting pose parameters that



Object Detection, Fig. 3 Bounding boxes and segmentation masks produced by Mask R-CNN. The system is trained to detect and segment 80 object categories from the COCO dataset. (Reproduced with permission from [21])

Object Detection, Fig. 4 Bounding boxes, segmentation masks, and joint positions produced by Mask R-CNN. The system is trained to detect, segment, and predict pose on the *person* categories from the COCO dataset. (Reproduced with permission from [21])

recover some of the quantization error, resulting in fine predictions. The progressive reduction in spatial resolution can also be used to efficiently construct a multi-scale pyramid representation from a single input image scale [23], which is more efficient than processing multiple input image scales independently. These networks also share nearly all the computation of the hierarchy of features across all object categories, rather than retraining a separate hierarchy for each one *vs.* rest binary classification models, one for each object category. The shared computation enables such networks to scale to thousands of object categories; the marginal per category cost grows linearly but is small relative to the computation shared between categories. Finally, typical convolutional networks can be accelerated both algorithmically via fast convolution methods and with specialized hardware that makes extensive use of parallel computation (e.g., GPUs).

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 still rely on thousands of training examples per category in order to achieve reasonable results.

A popular benchmark for object detection is the COCO object detection challenge [24]. The goal of the challenge is to detect objects from 80 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 each year by a significant amount.

Detection methods that segment objects are also steadily improving and are now deployed for various applications, particularly in augmented and virtual reality scenarios. Figure 3 shows example output of Mask R-CNN [21]. Models that additionally predict human joint locations, in addition to bounding boxes and segmentation masks, are also useful in many applications including video conferencing systems that can automatically keep participants framed within the video steam. Mask R-CNN can be extended to predict human pose, as illustrated in Fig. 4.

References

1. LeCun Y, Boser B, Denker JS, Henderson D, Howard RE, Hubbard W, Jackel LD (1989) Backpropagation applied to handwritten zip code recognition. Neural Comput 1(4):541–551
2. Amit Y (2002) 2D object detection and recognition: models, algorithms and networks. MIT Press, Cambridge
3. Felzenszwalb P, Huttenlocher D (2005) Pictorial structures for object recognition. Int J Comput Vis 61(1):55–79
4. Fergus R, Perona P, Zisserman A (2003) Object class recognition by unsupervised scale-invariant learning. In: IEEE CVPR 2003
5. Amit Y, Trouvé A (2007) POP: patchwork of parts models for object recognition. Int J Comput Vis 75(2):267–282
6. Jin Y, Geman S (2006) Context and hierarchy in a probabilistic image model. In: IEEE CVPR 2006
7. Rowley HA, Baluja S, Kanade T (1998) Neural network-based face detection. IEEE Trans Pattern Anal Mach Intell 20(1):23–38
8. Viola P, Jones MJ (2004) Robust real time face detection. Int J Comput Vis 57(2):137–154
9. Fleuret F, Geman D (2001) Coarse-to-fine face detection. Int J Comput Vis 41(1–2):85–107
10. Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. In: IEEE CVPR 2005
11. 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
12. Reiss T (1993) Recognizing planar objects using invariant image features. Springer, Berlin
13. Riesenhuber M, Poggio T (2000) Models of object recognition. Nat Neurosci 3(11s):1199–1204
14. 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
15. Lowe D (2004) Distinctive image features from scale-invariant keypoints. Int J Comput Vis 60(2):91–110
16. 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(2):117–140
17. Krizhevsky A, Sutskever I, Hinton G (2012) Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems
18. Girshick R, Donahue J, Darrell T, Malik J (2014) Rich feature hierarchies for accurate object detection and semantic segmentation. In: IEEE CVPR 2014
19. Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu CY, Berg AC (2016) SSD: single shot multibox detector. In: ECCV 2016
20. Ren S, He K, Girshick R, Sun J (2015) Faster R-CNN: towards real-time object detection with region proposal networks. In: Advances in neural information processing systems
21. He K, Gkioxari G, Dollár P, Girshick R (2017) Mask R-CNN. In: ICCV 2017
22. Ullman S (1996) High-level vision. MIT Press, Cambridge, MA
23. Lin TY, Dollár P, Girshick R, He K, Hariharan B, Belongie S (2017) Feature pyramid networks for object detection. In: IEEE CVPR 2017
24. Lin TY, Maire M, Belongie S, Hays J, Perona P, Ramanan D, Dollár P, Zitnick CL (2014) Microsoft COCO: common objects in context. In: ECCV 2014

Object Extraction

▸ Interactive Segmentation

Object Models

▸ Model-Based Object Recognition: Traditional Approach

Object Motion Blur

▸ Motion Blur

Object Parameterizations

▸ Model-Based Object Recognition: Traditional Approach

Object Recognition

▸ Line Drawing Labeling

Object Representations

▸ Model-Based Object Recognition: Traditional Approach

Object Segmentation

▸ Semantic Image Segmentation: Traditional Approach

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.

Development of obstacle detection systems has accelerated in the past decade, due to rapid progress in sensors, embedded computing, and growing markets for intelligent vehicles. The rising commercial importance of automated driver assist systems (ADAS) and self-driving cars and trucks has propelled development of obstacle detection systems for vehicles on roads [1, 2]. This includes low-cost sensors, abilities to detect

and predict the paths of moving obstacles, and abilities to classify the types of obstacles (e.g., fence, pedestrian, bicycle, car) to better understand how to react to them. The potential commercial importance of unmanned air vehicles (UAVs) similarly fuels development of small, lightweight obstacle detection systems, as well as other techniques, to allow such vehicles to fly safely in shared airspace [3, 4]. At the most remote end of the application spectrum, obstacle detection has become important in planetary exploration for rovers and landers [5, 6].

Sensors for Obstacle Detection

An early introduction to a broad range of sensors for obstacle detection is included in [7]. Three-dimensional perception plays a central role in obstacle detection, because the geometry of the world is a major factor in determining what constitutes an obstacle. Three-dimensional sensors include active ranging techniques, like lidar [8–10], radar [11], and sonar [12], and passive ranging techniques, like stereovision [13] and structure from motion [14]. 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. The very wide range of illumination present in outdoor scene has been a challenge for selecting cameras both for human viewing and for automated obstacle detection; this is being addressed by advances in high dynamic range imagers [15, 16].

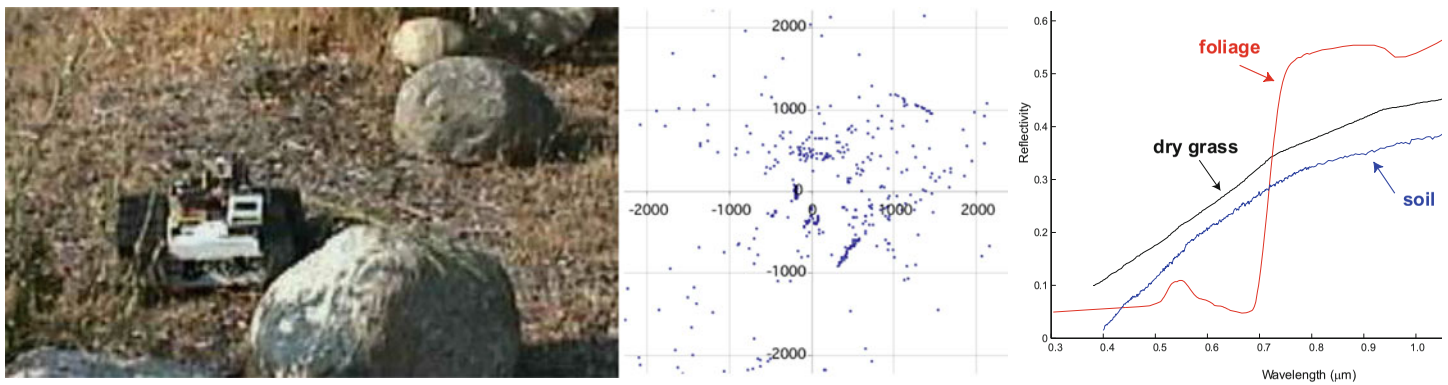
Mechanical properties like the stiffness of objects or terrain often are important to obstacle detection, as well as geometric properties. This requires sensors that directly or indirectly give insight into mechanical properties of the scene. These can be proprioceptive sensors, like inertial sensors, 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 lidar [17] or multispectral imagery [18] 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 [19, 20]. For example, fog, smoke, and night operation may require active sensors or thermal infrared cameras, while thick dust or heavy precipitation may require radar (Fig. 2). 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 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 lidar, for example, surface normals, range discontinuities, and height aboveground 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. Performing obstacle detection in sensor space minimizes computational cost and obstacle detection latency, which is important for



Obstacle Detection, Fig. 1 Left: small robot in sparse grass, carrying a single-axis scanning lidar. Middle: range data from the lidar (scale in millimeters), with lidar at the origin. The two large rocks to the right of the robot are apparent as linear structure among the scattered returns from vegetation. Right: spectral reflectivity vs. wavelength (in microns) of soil, dry grass, and green foliage, showing the high contrast of green foliage between visible and near-infrared bands

Obstacle Detection, Fig. 2 Smoke seen with visible spectrum and thermal infrared cameras; thermal infrared penetrates smoke and fog far better than visible wavelengths

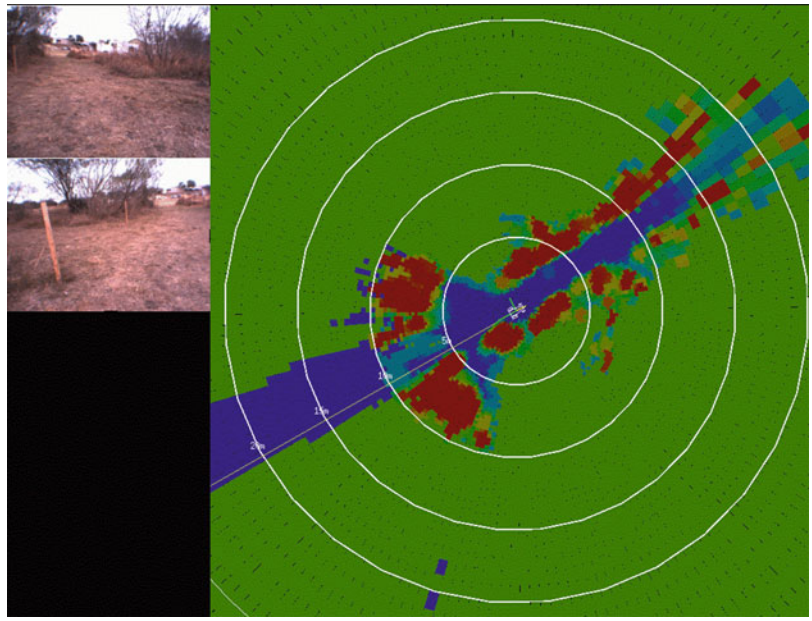
vehicles with limited computational resources or that move very fast and must react very quickly [21, 22].

An alternative to working in image space is to work in a *map space*. These are often expressed in a Cartesian reference frame that is either vehicle-centered or has a fixed origin in the world; they can also be expressed in polar coordinates with a vehicle-centered origin. Map representations can simply accumulate raw data from many sensors or many points in time, such as point clouds from range imaging sensors like lidar or stereovision. Alternatively, they may be discrete grids that are used to summarize range data over time, such as elevation maps and 2-D or 3-D occupancy grids. Elevation maps record the height of the ground surface in each cell of a discretized map of the ground plane; multiple heights or intervals of heights may also be recorded for each map cell in 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; in 3-D, they are often called *voxel maps*. Concepts of height maps and voxels have also been combined in the same system [24, 25]. Important issues for map data structures are that they allow rapid access to and update of information in each cell, rapid access to neighborhoods, and efficient use of memory for mapping large regions, including sparse representations. Trees, allocatable map tiles, voxel hashing, and multi-resolution variants on these themes are examples of techniques used to address these issues [23, 24, 26, 27] (Fig. 3).

Shape of the raw sensor uncertainty distribution also can drive the choice of world model representation, particularly for sensors with wide beams (leading to polar representations) and for passive triangulation from images, where inverse range parameterizations can be useful, particularly for image space representations [21]. Robot-centered, 2-D polar-perspective maps are another inverse range representation, where the radial axis of 2-D polar maps is parameterized with inverse range to match the spatial and range resolution characteristics of stereovision [28].

Geometric representations fit geometric primitives, like line segments or polygons for 2-D maps and planar surfaces or bounding boxes for 3-D maps, to the raw sensor data or to segmented gridded data. These are less commonly used for obstacle detection than for visualization or for strictly mapping purposes, but they have often been used as compact world models for motion planning algorithms. Other methods of compressing raw point cloud data into a more compact representation include clustering the points and fitting the clusters with a mixture of Gaussian distributions [29].

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



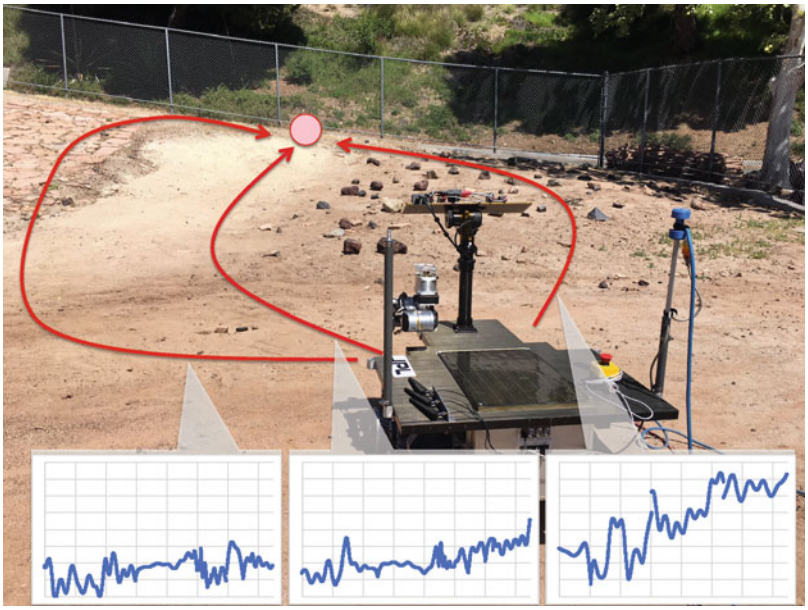
Obstacle Detection, Fig. 3 Hybrid 2-D cost map [23], with a Cartesian map closest to the robot and a polar map further away to reflect resolution characteristics of stereovision. White rings are 5 m intervals. Blue codes low cost, while red is high cost, revealing corridor between the bushes in the upper left image

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 [30]. 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. Common approaches to this include recursive filtering of Gaussian models of uncertainty in range or height measurements [31], and updating map cell occupancy inferences is the use of occupancy grids to estimate the probability that map cells contain obstacles. Efforts to improve elevation mapping have used Gaussian process [34] and Markov random field formulations [35]. Several techniques have been explored to extend occupancy grids to improve their spatial fidelity and/or statistical model, including normal distributions transform occupancy maps (NDT-OM) [32] and Hilbert maps [33]. Gaussian mixture models have been applied to temporal fusion of depth maps [36].

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 lidars 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 [37, 38], the energy cost of a robot path [39], or other metrics [40] (Fig. 4).



Obstacle Detection, Fig. 4 Predictions of driving energy for three different vehicle paths encountering different slopes and predicted slippage [39]

Algorithms and Systems 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 efficient path planning. There are many approaches to this, which depend on the application context. The summary here focuses on ground vehicles to illustrate the diversity of issues and approaches involved; methods tailored for maritime vehicles, flying vehicles, and robot manipulators can be found in the relevant literature.

Among the simplest and oldest methods are those that apply to indoor ground vehicles with 2-D world models. Off-road vehicles require more complex methods to cope with the 2.5-D world model and the spectrum between rigid and compliant obstacles. On-road vehicles face complexities of other traffic, high speed, and a wide range of illumination, weather, and seasonal conditions that stress the need to address obstacle detection algorithms in the context of a multisensor autonomous driving system.

For indoor vehicles with 2-D world models, binarized occupancy grids and analogous data structures can be the obstacle map. The simple abstraction of treating the vehicle as circular enables expanding the obstacles by the radius of the vehicle to allow motion planning algorithms to treat the vehicle as a point in the 2-D ground plane. Similar concepts have been applied to stereo disparity maps to allow motion planning with image space representations for ground and air vehicles [21, 41].

For off-road vehicles, the world model must be at least 2.5-D, the degree of rigidness versus compliance of materials in the environment is significant, and the effects of vehicle dynamics are important at high speed. For Mars rovers, which move very slowly in desert terrain with no vegetation or moisture, obstacle detection algorithms can test for ground clearance by "kinematic settling" of an articulated vehicle model onto an elevation map of the terrain [5]. Limited onboard computational power may lead to using such algorithms in a hierarchical scheme that uses simpler methods to conduct initial terrain triage, for example, by fitting planes to vehicle-sized patches of the elevation map to distinguish regions that are very smooth or very rough from those that need more careful analysis of ground clearance. Sand dunes are compliant terrain that increase wheel slippage and can get the vehicle stuck. Research is in progress on

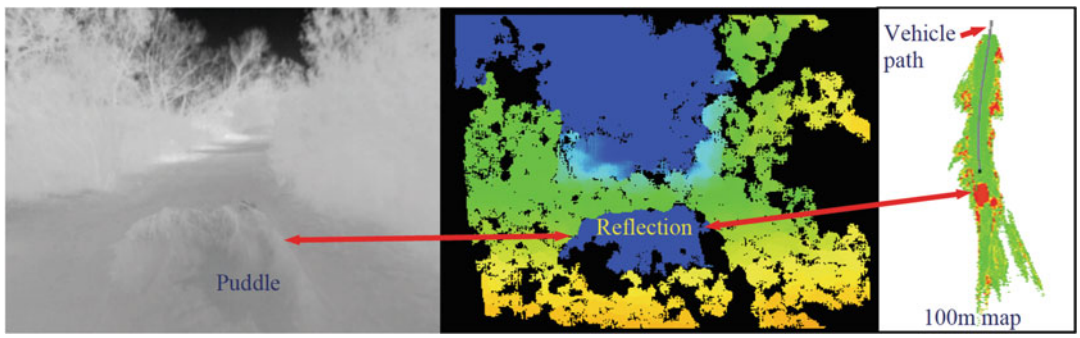
terrain classification methods that use onboard cameras to allow rovers to discriminate this terrain type from rigid terrain like bedrock [43].

Off-road vehicles on Earth face a much greater variety of terrain types and a wider range of illumination and atmospheric conditions that degrade sensor performance. Besides soft sand, terrain types include other compliant materials like vegetation, mud, and water bodies. A variety of sensors and detection algorithms have been explored for identifying these terrain types, including image classification, reasoning about 3-D point cloud statistics, detecting polarized reflections, and reasoning about temperature of the scene as revealed by thermal images [19, 44, 45] (Fig. 5). Manmade obstacles like wire may be present, which are thin and hard to see; specialized analysis of 3-D point cloud data has been used to detect these [44]. Holes in the ground of various sizes, including potholes and ditches, are collectively referred to as *negative obstacles*; these are hard to detect because they project to a small area in image space. Under some conditions, visible portions of the interiors of negative obstacles have temperatures that contrast with the surrounding terrain; this has led to thermal cues being included in specialized detectors for negative obstacles [47]. A variety of self-supervised and imitation learning algorithms have been explored for enabling vehicles to learn about terrain types from their own experience and from observing human driving examples [48,49]. Dynamic effects of vehicle interaction with off-road terrain types is still a fairly immature topic, especially for high-speed driving.

On-road driving generally has simpler vehicle-terrain interaction than off-road driving, but it has more demanding requirements for coping with high speed; navigating among moving objects that include motor vehicles, bicycles, pedestrians, and other objects; being robust to degraded seeing conditions (night, bad weather); and doing all of this while respecting many rules of the road. This requires detecting and classifying objects that may be in the path of the vehicle, tracking moving objects, and semantic labeling and segmentation of the scene as a whole. Cameras and lidars are primarily used for these tasks. Learning-based algorithms using deep convolutional neural networks have become very prominent in this domain. Survey articles provide a good introduction to this complex problem [50].

Open Problems

The complexity of understanding the on-road driving environment and the level of reliability required in that environment continues to have open problems, which strongly overlap other computer vision problems like object recognition, object tracking, and semantic segmentation. Off-road driving also presents difficult challenges for perceiving material types



Obstacle Detection, Fig. 5 Potential water hazard detection based on geometric reasoning about reflections in image and stereovision-based point cloud data [19]. Left: thermal image from a stereo image pair, with a puddle in the foreground. Center: false color depth map from stereo, with anomalous large range measured to the reflection. Right: cost map for 100 m ahead of the vehicle, with the puddle properly placed in the map after reasoning about range around its borders

and inferring whether they constitute obstacles that must be avoided or compliant terrain that can be driven over at some appropriate speed. There currently is a heavy reliance on 3-D perception with lidar for obstacle detection, but there is strong interest in being able to do more with cameras, because of potential for reducing cost and reducing signals emitted by the sensors. Poor seeing conditions (e.g., night, bad weather) present challenges that are not fully addressed. Obstacle detection in these domains is very computationally intensive; there is a perpetual interplay between development of more efficient algorithms and higher performance onboard computing systems to address this challenge.

References

1. Kukkala VK, Tunnell J, Pasricha S, Bradley T (2018) Advanced driver-assistance systems. IEEE Consumer Electronics Magazine
2. Yu X, Marinov M (2020) A study on recent developments and issues with obstacle detection systems and automated vehicles. Sustainability 12(8)
3. Kanellakis C, Nikolakopoulos G (2017) Survey on computer vision for UAVs: current developments and trends. J Intell Robot Syst 87(1):141–168
4. Bloise N, Primatesta S, Antonini R, Fici GP, Gaspardone M, Guglieri G, Rizzo A (2019) A survey of unmanned aircraft system technologies to enable safe operations in urban areas. In: International conference on unmanned aircraft systems, Atlanta, June 2019
5. Otsu K, Matheron G, Ghosh S, Toupet O, Ono M (2019) Fast approximate clearance evaluation for rovers with articulated suspension systems, July 2019
6. Robertson EA (2017) Synopsis of precision landing and hazard avoidance (PL&HA) capabilities for space exploration. AIAA guidance, navigation, and control conference, Grapevine, Jan 2017
7. Everett HR (1995) Sensors for mobile robots: theory and application. A. K. Peters, Wellesley
8. Warren ME (2019) Automotive LIDAR technology. In: Symposium on VLSI circuits, June 2019
9. Zhao F, Jiang H, Liu Z (2019) Recent development of automotive LIDAR technology, industry and trends. In: Proceedings of SPIE, vol 11179: eleventh international conference on digital image processing, Aug 2019
10. Nunes-Pereira EJ, Peixoto H, Teixeira J, Santos J (2020) Polarization-code material classification in automotive LIDAR aiming at safe autonomous driving implementations. Appl Opt 59(8)
11. Steinbaeck J, Steger C, Holweg G, Drumi N (2017) Next generation radar sensors in automotive sensor fusion systems. In: IEEE conference on sensor data fusion: trends, solutions, applications, Oct 2017
12. Song Y, Liao C (2016) Analysis and review of state-of-the-art automatic parking assist system. In: IEEE international conference on vehicular electronics and safety, July 2016
13. Tippetts B, Lee DJ, Lillywhite K, Archibald J (2016) Review of stereo vision algorithms and their suitability for resource-limited systems. J Real-Time Image Process 11(1):5–25
14. Saputra M, Markham A, Trigoni N (2018) Visual SLAM and structure from motion in dynamic environments: a survey. ACM Comput Surv 51(2)
15. Velichko S, Johnson S, Pates D, Silsby C, Hoekstra C, Mentzer R, Beck J (2017) 140 dB dynamic range subelectron noise floor image sensor. In: International image sensor workshop, June 2017
16. Sakano Y et al. (2020) A 132dB single-exposure-dynamic-range CMOS image sensor with high temperature tolerance. In: IEEE international solid-state circuits conference, Feb 2020
17. 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
18. 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
19. Rankin A et al. (2011) Unmanned ground vehicle perception using thermal infrared cameras. In: Proceedings of SPIE, vol 8045. Unmanned systems technology XIII, 2011
20. Judd K, Thornton M, Richards A (2019) Automotive sensing: assessing the impact of fog on LWIR, MWIR, SWIR, visible, and LIDAR imaging performance. In: Proceedings of SPIE, vol 11002: infrared techology and applications XLV, May 2019
21. Matthies L, Brockers R, Kuwata Y, Weiss S (2014) Stereo vision-based obstacle avoidance for micro air vehicles using disparity space. In: IEEE international conference on robotics and automation, May 2014
22. Barry AJ, Tedrake R (2015) Pushbroom stereo for high-speed navigation in cluttered environments. In: IEEE international conference on robotics and automation, May 2015
23. Bajracharya M, Howard A, Matthies LH, Tang B, Turmon M (2009) Autonomous off-road navigation with end-to-end learning for the LAGR program. J Field Robot 26(1)
24. Bajracharya M, Ma J, Howard A, Matthies L (2012) Real-time 3D stereo mapping in complex dynamic environments. In: Workshop on semantic perception, mapping, and exploration, held in conjunction with IEEE international conference on robotics and automation, May 2012

25. Garrote L, Premebida C, Silva D, Nunes U (2018) HMAPS – hybrid heigh-voxel maps for environment representation. IROS, Oct 2018
26. Hornung A, Wurm KM, Bennewitz M, Stachniss C, Burgard W (2013) OctoMap: an efficient probabilistic 3D mapping framework based on octrees. Auton Robot 34:189–206
27. Xu Y, Wu Y, Zhou H (2018) Multi-scale voxel hashing and efficient 3D representation for mobile augmented reality. In: IEEE/CVF CVPR workshops, June 2018
28. 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
29. Srivastava S, Michael N (2019) Efficient, multi-fidelity perceptual representations via hierarchical Gaussian mixture models. IEEE Trans Robot 35(1)
30. Moravec H, Elfes AE (1985) High resolution maps from wide angle sonar. In: Proceedings of the IEEE international conference on robotics and automation (ICRA), pp 116–121
31. Engel J, Stuckler J, Cremers D (2015) Large-scale direct SLAM with stereo cameras. In: IEEE/RSJ international conference on intelligent robots and systems (IROS), Sept 2015
32. Saarinen JP, Andreasson H, Stoyanov T, Lillienthal AJ (2013) 3D normal distributions transform occupancy maps: an efficient representation for mapping in dynamic environments. Int J Robot Res 32(14)
33. Ramos F, Ott L (2015) Hilbert maps: scalable continuous occupancy mapping with stochastic gradient descent. In: Robotics science and systems conference, July 2015
34. Kjaergaard M, Bayramoglu E, Massaro AS, Jensen K (2011) Terrain mapping and obstacle detection using Gaussian processes. In: 10^{th} international conference on machine learning and applications
35. Tse R, Ahmed NR, Campbell M (2015) Unified terrain mapping model with Markov random fields. IEEE Trans Robot 31(2)
36. Cigla C, Brockers R, Matthies L (2017) Gaussian mixture models for temporal depth fusion. In: IEEE winter conference on applications of computer vision, Mar 2017
37. 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
38. Liu J, Jayakumar P, Stein JL, Ersai T (2016) A study on model fidelity for model predictive control-based obstacle avoidance in high-speed autonomous ground vehicles. Int J Veh Mech Mobility 54(11)
39. Higa S, Iwashita Y, Otsu K, Ono M, Lamarre O, Didier A, Hoffmann M (2019) Vision-based estimation of driving energy for planetary rovers using deep learning and terramechanics. IEEE Robot Autom Lett 4(4)
40. (2010) Special issue on vehicle-terrain interaction for mobile robots. Int J Field Robot 27(2)
41. Otte M, Richardson S, Mulligan J, Grudic G (2009) Path planning in image space for autonomous robot navigation in unstructured outdoor environments. J Field Robot 26(2)
42. Otsu K, Matheron G, Ghosh S, Toupet O, Ono M (2019) Fast approximate clearance evaluation for rovers with articulated suspension systems. J Field Robot 37:768–785
43. Rothrock B, Papon J, Kennedy R, Ono M, Heverly M (2016) SPOC: deep learning-based terrain classification for Mars rover missions. AIAA space forum, Sept 2016
44. 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
45. 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
46. Rankin A, Huertas A, Matthies L, Bajracharya M, Assad C, Brennan S, Bellutta P, Sherwin G (2011) Unmanned ground vehicle perception using thermal infrared cameras. In: Proceedings of SPIE, vol 8045: unmanned systems technology XIII, Apr 2011
47. Matthies L, Rankin A (2003b) Negative obstacle detection by thermal signature. In: Proceedings of the IEEE/RSJ conference on intelligent robots and systems (IROS)
48. (2009) Special issue on LAGR program. Int J Field Robot 26(1)
49. Silver D, Bagnell JA, Stentz A (2010) Learning from demonstration for autonomous navigation in complex unstructured terrain. Int J Robot Res 29(12)
50. Yurtsever E, Lambert J, Carballo A, Takeda K (2020) A survey of autonomous driving: common practices and engineering technologies. IEEE Access 8:58443–58469

Occam's Razor

▸ Regularization

Occlusion

Sudipta N. Sinha
Microsoft Research, Redmond, WA, USA

Related Concepts

▸ Occlusion Detection
▸ Occlusion Handling

Definition

In computer vision, the term *occlusion* refers to the phenomenon which occurs when a portion of a 3D scene is not visible from the camera or another sensor. Visual rays from 3D points in such portions of the scene are obstructed or blocked from reaching the camera by another non-transparent surface or object that lies between the obstructed region and the image plane of the camera. Occlusion is a consequence of the nature of 3D projections and arises from the depth ordering of surfaces in the scene from a particular viewpoint. The region which is occluded, i.e., not visible from the camera, is often referred to as the *occluded region*, whereas the obstructing objects or surfaces are referred to as *occluders*. Finally, the visible edge or boundary of the occluder or occluding surface is often referred to as the *occlusion boundary*.

Occlusions in Object Tracking

Occlusion occurs naturally in most 3D scenes where multiple objects and complex geometric shapes are present in arbitrary configurations or when the scene is dynamic and objects deform or move within the scene. The extent of the occlusion may be higher in cluttered scenes compared to scenes which contain very few objects. The degree of occlusion may also depend on the camera viewpoint in the scene. For example, in an open outdoor scene, an upright camera slightly above ground level may have significantly more occlusion compared to a camera looking downward into the scene from a greater height.

The presence of occlusion leads to a wide range of challenges for a class of computer vision techniques that solve various visual recognition tasks, namely, object detection, object tracking, segmentation, and visual recognition. Let us briefly discuss why occlusions make object tracking challenging. Figure 1a–c shows a person in three video frames from a single camera. The person's face is occluded by the magazine in two of the frames. In this case, the goal is to track the person's face. Specifically, the tracker should estimate the corners of a 2D rectangle that fits around the face on each frame, and we would like these predictions to be robust to occlusion as shown in Fig. 1d–f.

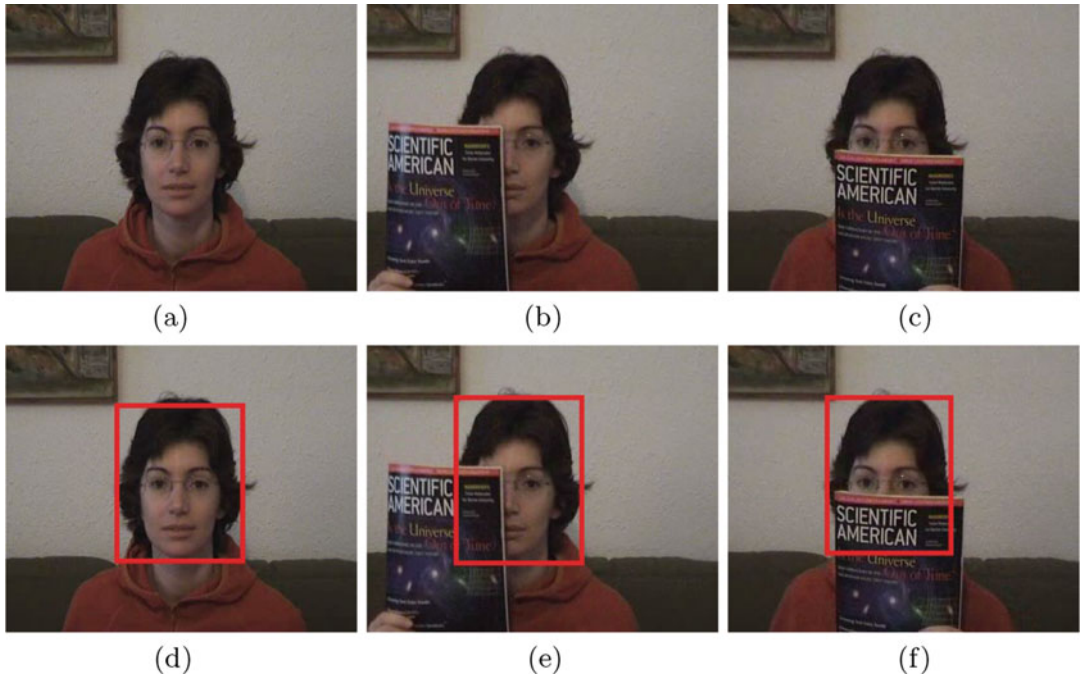
Most object tracking approaches construct and maintain an adaptive appearance model of the object being tracked. If the appearance model is global, i.e., represents the appearance of the complete object, a tracking approach that uses such a model would be more susceptible to tracking failure due to occlusions. Note that the appearance or shape of the occluder cannot be assumed to be known and the tracking algorithm cannot assume any knowledge of which parts of the person's face or to what extent it will be occluded.

To solve the problem more robustly in the presence of occlusion, a more flexible appearance model is typically used, one which constructs and adaptively maintains multiple redundant appearance representations of different parts of the object. The basic principle is that when the object is occluded, it will be assumed that at least some of its parts are visible and this will allow the detector or tracker to find the object under a modest degree of occlusion.

Occlusions may cause object trackers to lose track of an object in a long sequence. This may in fact be inevitable if the object is completely occluded behind some other scene objects and disappears from the camera view. However, when the object is visible again, an occlusion-aware tracking approach will redetect the object and recognize it to be the same object instance observed before and tracking can be reestablished.

Occlusion in Stereo and Optical Flow Estimation

We now discuss how occlusion presents some challenges in low-level image correspondence recovery tasks such as stereo matching and optical flow. In stereo matching, the goal is to estimate a dense set of pixel correspondences in two or more images where the scene is assumed to be rigid. This is possible, either with multiple

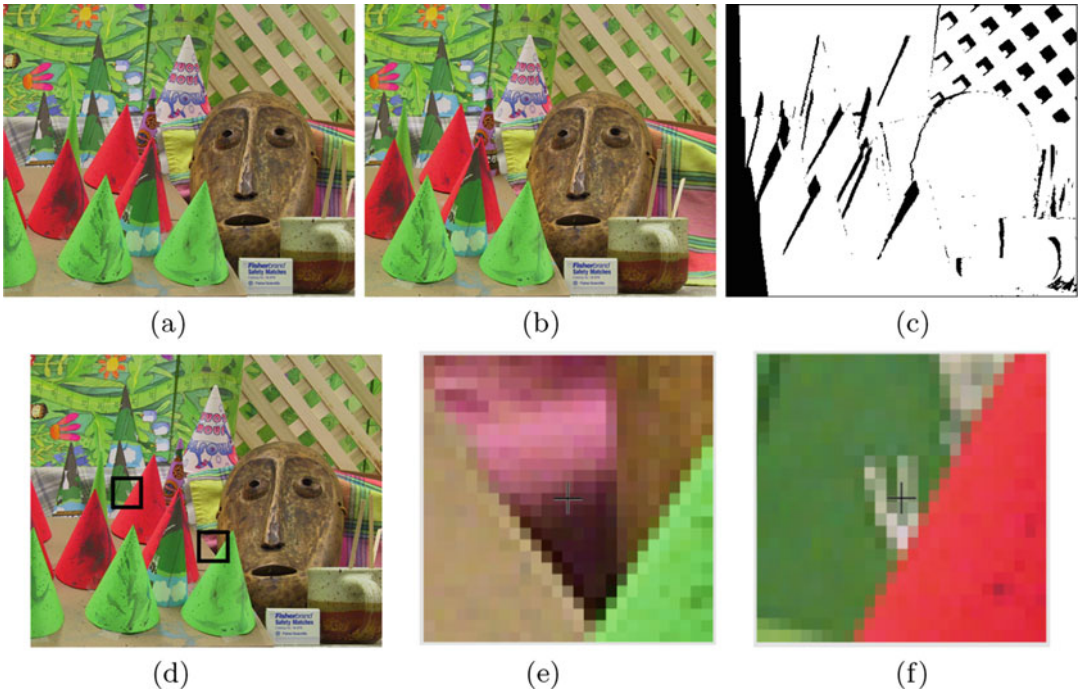


Occlusion, Fig. 1 Occlusion creates some challenges in object tracking. (**a**) A frame from the FACEOCC1 sequence showing a person, taken from the visual tracking benchmark [10]. (**b**) A different frame from the video where the person's face is considerably occluded. (**c**) Another frame where a different part of the person's face is occluded. (**d**–**f**) Object tracking results that can be expected on these images from a tracking technique which is robust to occlusion

cameras in the scene or using a single moving camera but by assuming the scene to be stationary. Similarly, in optical flow estimation, one also seeks a dense set of pixel correspondences in multiple images such as from consecutive frames of video. In optical flow, general pixel motion is allowed as would occur in a nonrigid scene or a scene with rigid objects that are moving independently.

Figure 2 shows an example of occlusion in the binocular stereo problem. The leftmost and center image depicts a pair of stereo-rectified images where the corresponding pixels lie on identical horizontal scanlines of the two images. Figure 2c depicts a binary mask where the white pixels in the mask image correspond to pixels in the left image that are visible in the right image whereas the black pixels in the mask image are not visible in the right image. This binary mask is often referred to as the *occlusion mask*.

Occlusion in the stereo matching setting arises from the effect of parallax and is associated with edges in the scene that form depth discontinuities, i.e., where the depth of pixels changes abruptly. While the fraction of occluded pixels in a stereo pair depends on the scene geometry and range of parallax, one typically observes a higher degree of occlusion when the baseline between the stereo cameras is increased. Scenes where occlusion occurs more frequently are typically more challenging for stereo matching. Estimating the disparity (or scene depth) of occluded pixels requires reasoning using monocular cues instead of stereo cues. Similarly, in the case of optical flow estimation, occlusion is associated with edges that form motion discontinuities, and computing optical flow for sequences with fast and large motion is typically more challenging as they are likely to produce more occluded pixels.



Occlusion, Fig. 2 Occlusion makes stereo matching more challenging. (**a**–**b**) The left and right image from CONES, a pair of rectified stereo images from the Middlebury stereo matching benchmark [5]. (**c**) The occlusion mask for the pixels in the left image. The white pixels in the mask are visible, i.e., non-occluded in the right image, whereas the black pixels are not visible, i.e., occluded in the right image. (**d**) The left image shown again with two occluded regions marked by black squares. (**e**–**f**) Images showing zoom-ins of the patches within the black squares

It is possible to explicitly model the occlusion phenomena in stereovision using geometric reasoning. Such stereo matching methods typically infer the disparity map in both left and right images [2, 7] and may involve explicit estimation of the left and right occlusion maps. Typically such techniques require multiple phases, where during the initial phase the occlusion maps are estimated whereas in the subsequent phases the estimated occlusion maps are used to adjust appropriate terms in the objective function in an occlusion-aware manner.

However, many stereo matching and optical flow estimation methods avoid explicitly modeling occlusions and instead treat the occurrence of occlusions as outliers. These methods implicitly deal with occlusion by either using robust cost functions in the algorithmic formulation or employing robust optimization procedures to infer the correspondence. This idea is quite effective in the multi-view stereo task when several overlapping viewpoints are available [9]. Most learning-based methods also model occlusions implicitly and rely on many examples in the training set to indirectly achieve robustness to occlusions.

Occlusion Reasoning and Scene Understanding

Reasoning about occlusions in a scene and detecting occlusion boundaries can provide indirect cues about the 3D scene geometry. It can also help to decompose the scene into a set of discrete layers which may aid 3D scene understanding. For example, it is possible to detect occlusion boundaries by performing background segmenta-

tion on a moving object in a stationary scene and then analyzing the edge and boundary statistics of the foreground or object segments over time. When the object has been observed to move within the scene for a sufficiently long time, the boundary statistics may reveal not just the location of the occlusion boundaries but distinct layers in the scene geometry (from the camera's viewpoint) and the relative ordering of these layers [6]. Finally, occlusion reasoning can also be performed on videos captured using multiple calibrated cameras with overlapping viewpoints. By segmenting moving objects from the respective background images in multiple views, it is possible to both infer the 3D shape of the stationary occluders in the scene and reconstruct the dynamic objects more accurately based on explicit geometric occlusion reasoning using a probabilistic formulation [3].

Several other approaches have also been investigated to solve the occlusion boundary detection problem in single images using supervised learning approaches [4], by utilizing low-level motion cues based on optical flow computation on video [8] and based on the detection of T-junctions in images [1].

References

1. Apostoloff N, Fitzgibbon A (2005) Learning spatiotemporal t-junctions for occlusion detection. In: CVPR, vol 2. IEEE, pp 553–559
2. Bleyer M, Rother C, Kohli P, Scharstein D, Sinha S (2011) Object stereo—joint stereo matching and object segmentation. In: CVPR. IEEE, pp 3081–3088
3. Guan L, Franco J-S, Pollefeys M (2007) 3d occlusion inference from silhouette cues. In: CVPR. IEEE, pp 1–8
4. Hoiem D, Stein AN, Efros AA, Hebert M (2007) Recovering occlusion boundaries from a single image. In: 2007 IEEE 11th international conference on computer vision. IEEE, pp 1–8
5. Scharstein D, Hirschmüller H, Kitajima Y, Krathwohl G, Nešić N, Wang X, Westling P (2014) High-resolution stereo datasets with subpixel-accurate ground truth. In: German conference on pattern recognition. Springer, pp 31–42
6. Schodl A, Essa I (2001) Depth layers from occlusions. In: CVPR, vol 1. IEEE, p I
7. Sun J, Li Y, Kang SB, Shum H-Y (2005) Symmetric stereo matching for occlusion handling. In: CVPR, vol 2. IEEE, pp 399–406
8. Sundberg P, Brox T, Maire M, Arbeláez P, Malik J Occlusion boundary detection and figure/ground assignment from optical flow. In: CVPR. IEEE, pp 2233–2240 (2011)
9. Vogiatzis G, Esteban CH, Torr PHS, Cipolla R (2007) Multiview stereo via volumetric graph-cuts and occlusion robust photo-consistency. IEEE Trans Pattern Anal Mach Intell 29(12):2241–2246
10. Wu Y, Lim J, Yang M-H (2013) Online object tracking: a benchmark. In: CVPR

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, Leuven, Belgium

Synonyms

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

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, 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, Leuven, 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. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)

ODS

▸ Omnidirectional Stereo

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

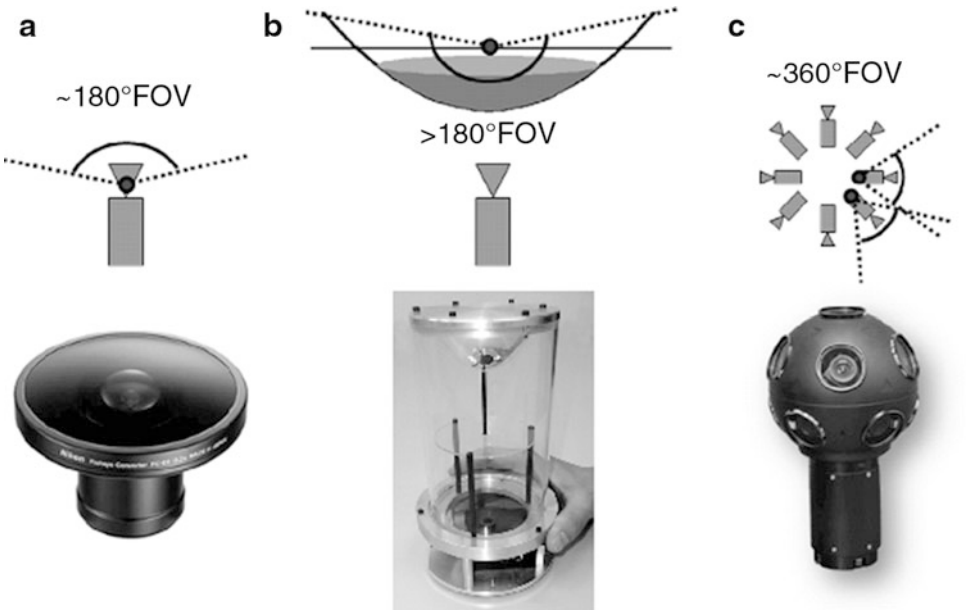
An omnidirectional camera (from *omni*, meaning all) is a camera with a 360-degree field of view in the horizontal plane or with a visual field that covers a hemisphere or (approximately) the entire sphere.

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



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

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.

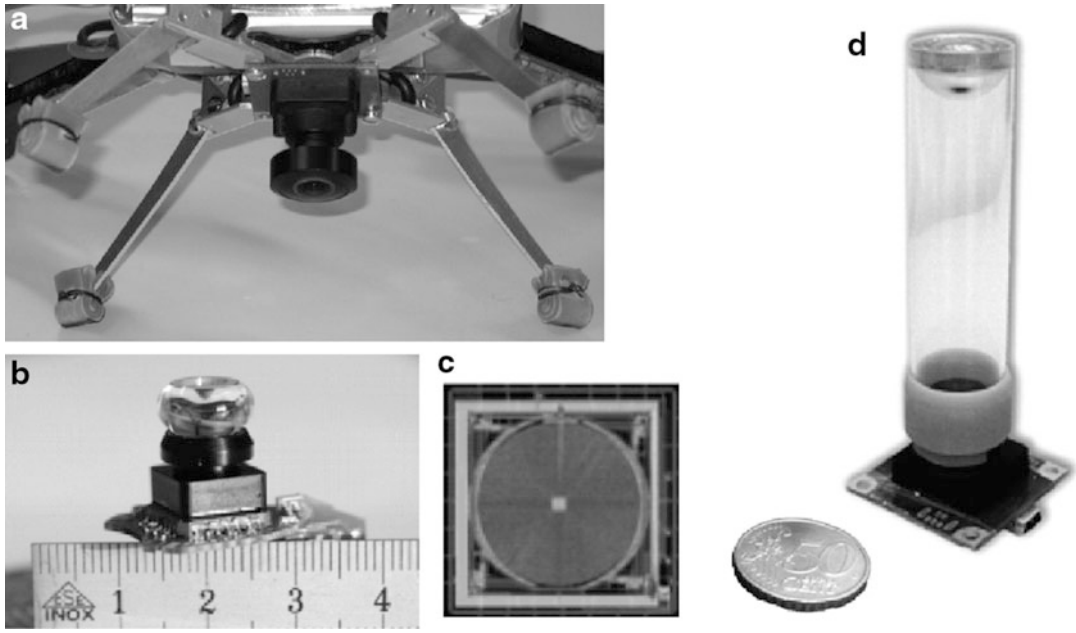
Theory

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. 3). 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 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. 4). 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. 4).

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



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

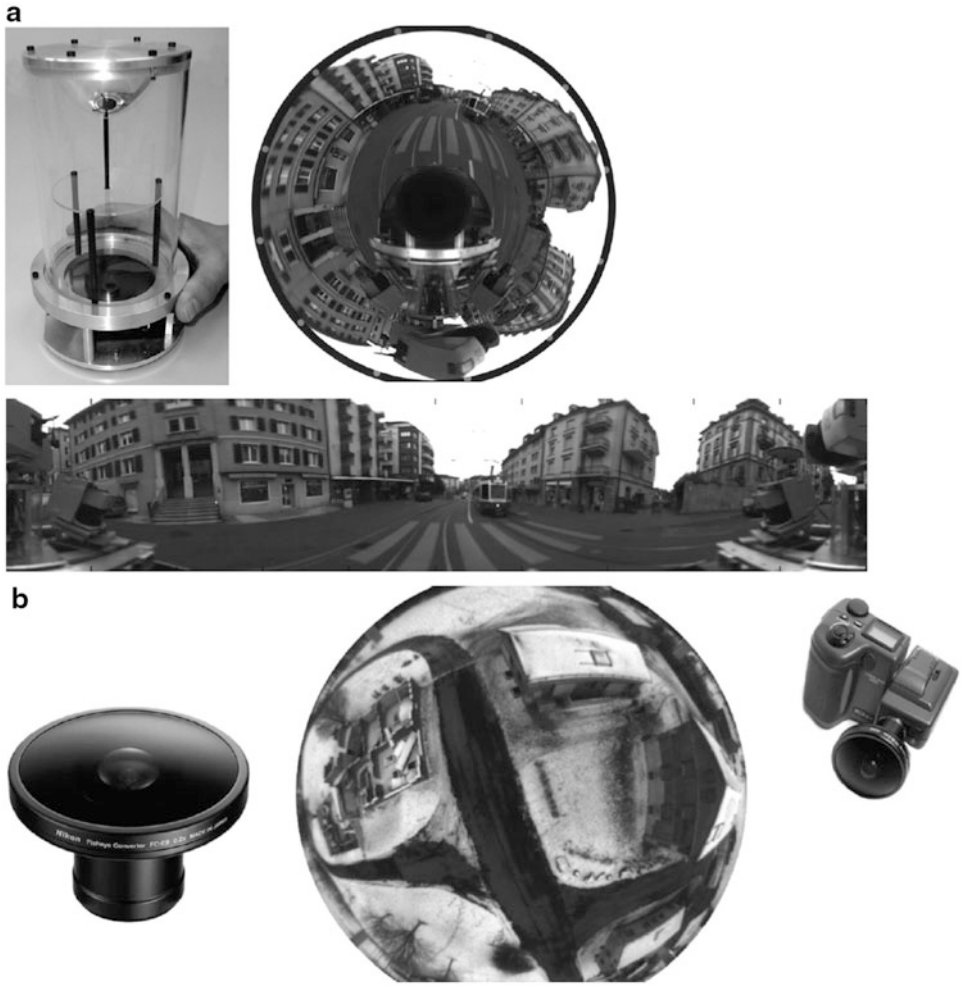
(Fig. 5). This is possible because, under the single view point constraint, every pixel in the sensed image measures the irradiance of the light passing through the viewpoint in one particular direction. When the geometry of the omnidirectional camera is known, that is, when the camera is calibrated, 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. 5, 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 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, 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



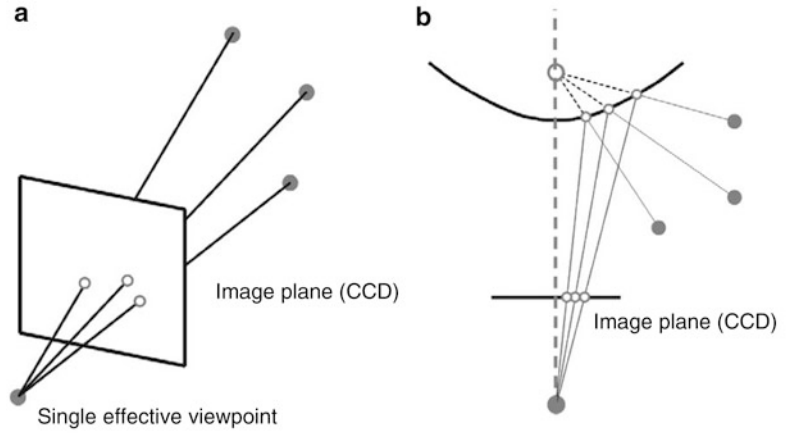
Omnidirectional Camera, Fig. 3 (**a**) A catadioptric omnidirectional camera using a hyperbolic mirror. The image is typically unwrapped into a cylindrical panorama. The field of view is typically 100° in elevation and 360° in azimuth. (**b**) Nikon fisheye lens FC-E8. This lens provides a hemispherical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

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].

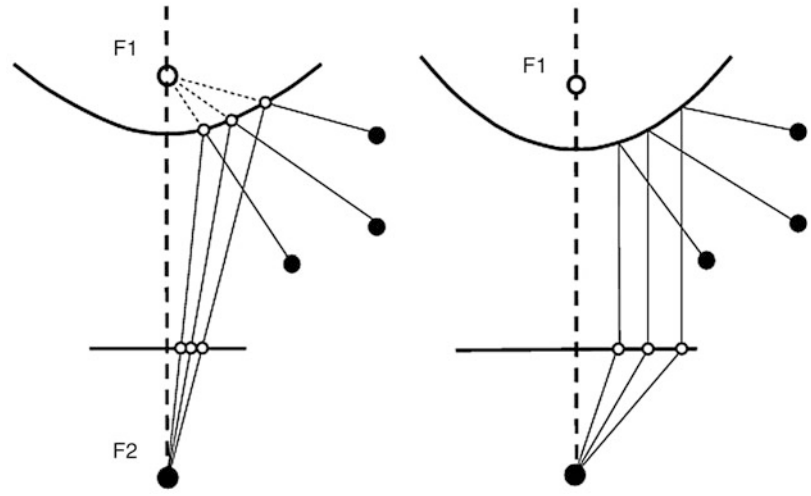
Conversely, the second model unifies both central catadioptric cameras and fisheye cameras under a general model also known as Taylor model. It was developed in 2006 by Scaramuzza et al. [10, 11] and has the advantage that both catadioptric and dioptric cameras can be described through the same model,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



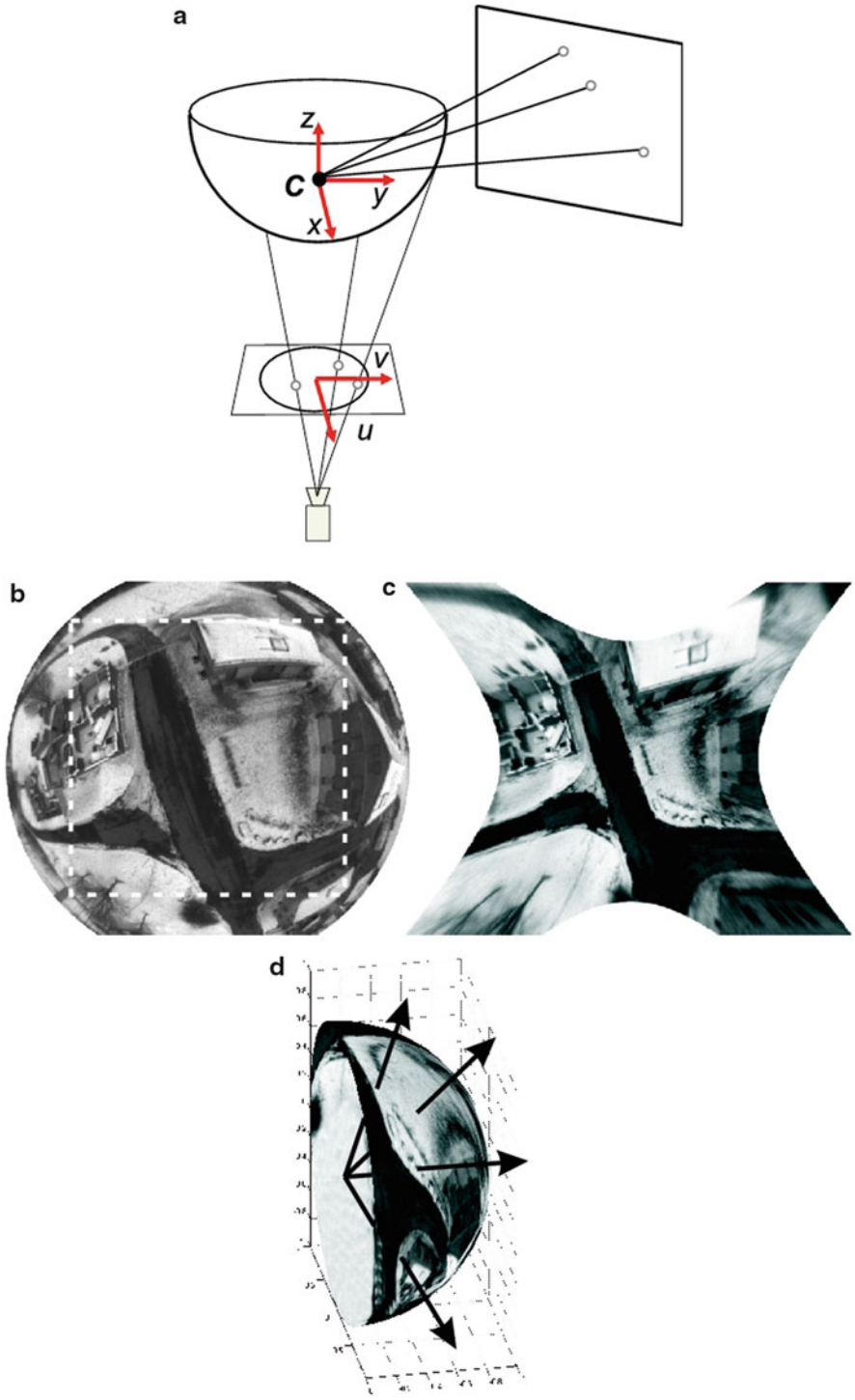
Omnidirectional Camera, Fig. 4 (**a**) and (**b**) Example of central cameras: perspective projection and catadioptric projection through a hyperbolic mirror. (**c**) and (**d**) Example of noncentral cameras: the envelope of the optical rays forms a caustic



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

the center of the sphere. This is summarized in Fig. 6.

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. 6). For convenience, we assume that the axis of symmetry of the mirror



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

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 $C_\varepsilon = (0,0,-\varepsilon)$; 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,

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

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

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

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

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

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 $\varepsilon = 0$ 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:

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

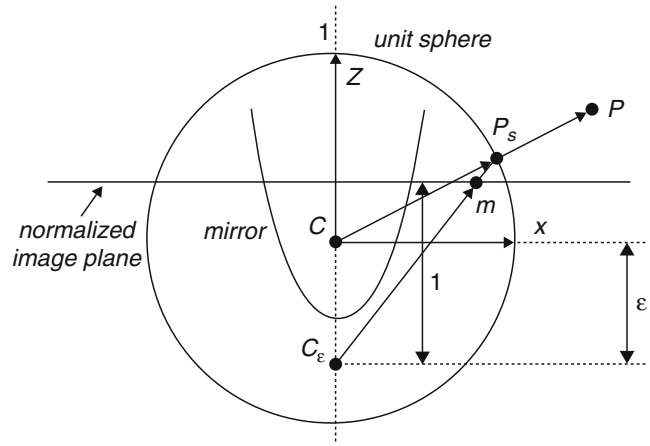
where $\rho = \sqrt{x_m^2 + y_m^2}$. As the reader may observe, the first-order term (i.e., a_1 ρ) of the polynomial is missing. This follows from the observation that the first derivative of the polynomial calculated at $\rho = 0$ must be null for both catadioptric and fisheye cameras (this is straightforward 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, 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, 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.



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

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. 7.

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. Found Trends Comput Graph 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. In: 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. In: 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 Stereo

Christian Richardt
University of Bath, Bath, UK

Synonyms

ODS; Omnistereo

Related Concepts

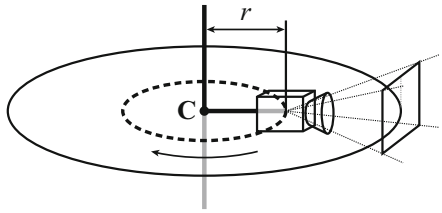
- ▸ Calibration of a Non-single Viewpoint System
- ▸ Image Stitching
- ▸ Omnidirectional Camera
- ▸ Video Mosaicing

Definition

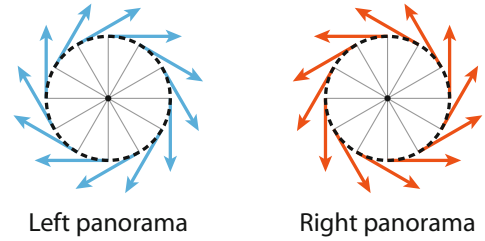
Omnidirectional stereo (ODS) is a type of multi-perspective projection that captures horizontal parallax tangential to a viewing circle. This data allows the creation of stereo panoramas that provide plausible stereo views in all viewing directions on the equatorial plane.

Background

The term "omnidirectional stereo" was first coined by Ishiguru et al. [1] in 1990, who used it in the context of autonomous mapping and exploration of an unknown environment. Their approach places a video camera on a rotating arm that is driven by a stepper motor (see Fig. 1). They then take vertical slit images from the sensor image for creating the left/right stereo panoramic views. As their primary goal is to map an environment, they use ODS images for depth estimation of scene points rather than display. They also present a binocular stereo method for depth estimation from two ODS images with a measure for direction-dependent uncertainty.



Omnidirectional Stereo, Fig. 1 Omnidirectional stereo images can be captured with a single camera mounted on an arm of length r that rotates about a point **C**. (Figure adapted from Ishiguru et al. [1])



Omnidirectional Stereo, Fig. 2 Omnidirectional stereo projection creates two panoramas, for the left and right views, using rays tangential to the viewing circle

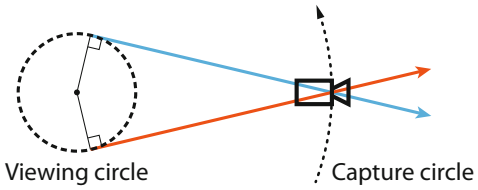
Before the term "omnidirectional stereo" became established with its current meaning, it was used more broadly for any stereo imaging system, which captures stereo panoramas with disparity, regardless of direction. Gluckman et al. [2], for example, created a real-time system consisting of two catadioptric cameras that are vertically displaced. These cameras capture omnidirectional views of an environment that therefore primarily differ by vertical disparity. While this works just as well for depth estimation, these vertical stereo panoramas are unsuitable for being viewed by humans. This is because our eyes are horizontally displaced and the human visual system thus expects horizontal disparity, not vertical disparity. In the following, the focus therefore lies on stereo panoramas with horizontal disparity.

Since the early days of ODS for mapping and robotics, the main application has shifted toward display on monitors, projection screens, and head-mounted virtual reality displays: Peleg et al. [3] popularized "omnistereo" panoramas with automatic disparity control, Richardt et al. [4] proposed improvements for creating high-quality, high-resolution stereo panoramas, and Anderson et al. [5] and Schroers et al. [6] developed state-of-the-art systems for capture and display of real-world virtual reality video. The benefit of ODS video is that time-varying stereo panoramas can easily be packed into a traditional video, which can be processed, stored, and transmitted on existing video streaming platforms just like any other video. This has established omnidirectional stereo as a widely supported format by video streaming pipelines and virtual reality displays.

Theory

Omnidirectional stereo is fundamentally a multi-perspective projection that is created from rays that are tangential to a *viewing circle* [3]. This projection can, for example, be captured by a slit camera that moves along a circular path and looks in the tangential direction. The slit images captured at different positions on the viewing circle are mosaicked into a panorama, producing the ODS projection. The two possible tangential directions give rise to the left and right panoramas, as illustrated in Fig. 2. The diameter of the viewing circle is usually chosen to be the average human interpupillary distance of 65 mm; however, other sizes are possible.

As we shall see, most ODS systems do not capture the ODS projection directly. The only implementation of a pair of rotating slit cameras is by Konrad et al. [7]. The key benefit of this approach is that the system directly captures the ODS projection without requiring computationally expensive processing. In addition, this system natively supports scenes containing nearby objects, occlusions, thin and repetitive textures, transparent and translucent surfaces, and specular and refractive objects, which remain challenging for other capture approaches. However, the prototype camera was limited to only five frames per second due to the fast rotation of the camera assembly that is required for operation.



Omnidirectional Stereo, Fig. 3 A single rotating camera can capture all rays tangential to the viewing circle, by rotating it on a larger circle. Note that this only works perfectly within the plane containing the viewing and capture circles; other rays are captured with vertical distortion. (Figure adapted from Anderson et al. [5])

Ishiguro et al. [1] first described and Peleg et al. [3] later popularized a single-camera approach that rotates a standard perspective camera on a larger *capture circle* that is concentric to the viewing circle. The camera needs to rotate a full 360 degrees to capture a complete ODS panorama, which assumes a static scene during the capture time. As illustrated in Fig. 3, the camera can simultaneously capture rays for both the left and the right stereo panorama. While this works perfectly in the 2D equatorial plane, out-of-plane rays introduce vertical distortion [5], and perspective cameras introduce perspective distortion. Richardt et al. [4] proposed techniques for correcting perspective distortion and for removing stitching artifacts using flow-based blending. Stabilization steps were also introduced that reduce drift from image alignment and enable handheld capture, where the camera's pose often deviates from the ideal position and orientation along the capture circle.

However, the rotating-camera approach is not practical for capturing ODS *videos*, as the camera needs to rotate a full 360 degrees for each video frame. For example, this would require very fast 30 revolutions per second for capturing ODS video with 30 frames per second. Peleg et al. [3] explored theoretical optical designs involving spiral mirrors and spiral lenses that would enable single-shot ODS capture and thus single-camera ODS video. However, these designs were never realized. The TORNADO system by Tanaka and Tachi [8] was the first practical implementation based on these designs. Elongated prism sheets deflect the light rays entering the cylindrical camera system and emulate 32 slit cameras that rotate over time. Inside the cylinder, a hyperboloidal mirror reflects light rays into a stationary video camera. The projections of the slit cameras into the video camera are separated in two different ways: (1) based on their spatial location in images or (2) using linear/circular polarizers. Aggarwal et al. [9] proposed a different approach that uses a carefully designed mirror with a "coffee filter" shape and reflects multiple slit camera images into the video camera without any additional optical elements. High-resolution simulations validate this concept, but manufacturing inaccuracies severely reduce the visual quality of real-world captures. Low visual quality and image resolutions are problems shared by all current catadioptric single-camera approaches [8, 9].

Approaches with multiple omnidirectional cameras improve some of the quality problems caused by complex optical elements. Chapdelaine-Couture and Roy [10] presented an approach for three or more fisheye cameras that are arranged in a regular polygon, with parallel viewing directions toward the sky. They observed that the planes spanned by the optical axes of adjacent pairs of cameras can be used for cutting their omnidirectional images and stitching them together with minimal disparity mismatch. However, this approach can only capture the upper hemisphere of any scene. Matzen et al. [11] proposed a system that uses two consumer spherical cameras, which have two 180-degree fisheye lenses each. The cameras are mounted side by side with a distance of 64 mm, and the captured imagery is sliced by the plane going through the cameras (orthogonal to the optical axes) and recombined to create the necessary left/right panoramas for ODS imagery. Their approach cancels vertical disparity at the seams using warping in image space and uses this disparity to correct horizontal disparity. As horizontal disparity is maximal in front of the cameras but zero along the camera baseline, this correction depends on the azimuth angle. In both cases, the resolution of the resulting ODS video was not sufficient for high-quality virtual reality experiences due to limited sensor resolutions [10, 11].

The highest visual quality and image resolution of ODS video has been demonstrated by approaches that use multiple video cameras spaced evenly on a circle, like a discretized version of the continuously rotating camera of earlier approaches [1, 3, 4]. Anderson et al. [5] pack 16 GoPro cameras with fisheye optics into a tightly packed ring of 28 cm diameter, demonstrating that this is a sweet spot between more cameras and a smaller ring diameter that reduces vertical distortion. Their work comprises a detailed analysis of the sources of vertical parallax and distortion and presents a flow-based stitching pipeline that produces temporally coherent ODS video, as available on YouTube, for example. Schroers et al. [6] also use 16 video cameras, but formulate their video pipeline based on continuous light field reconstruction from sparse samples. This naturally leads to a panoramic image formation model that allows computer-generated imagery to be composited with captured real-world content. They also provide an analysis of the minimum visible depth that can be stitched or reconstructed, based on the number of cameras in the rig and their field of view.

Application

The initial application for omnidirectional stereo was for mapping and exploring environments using robots [1]. In this scenario, a single rotating camera provides a low-cost and effective solution for reconstructing a given static scene. Ishiguro et al. [1] demonstrate how to estimate the depth of points from a single ODS image but also from a binocular pair of ODS images. They further show that the accuracy of depth estimation depends on the direction relative to the camera baseline, with parallel directions having the highest uncertainty. This uncertainty can be decreased by integrating multiple observations in the form of local maps into a single global map.

Since then, omnidirectional stereo has become the de facto standard projection for stereo panoramas that are used for virtual reality photography and video. Google's Cardboard Camera app, for example, is based on the principles described by Richardt et al. [4] for creating high-quality, high-resolution stereo panoramas. Multiple independently developed systems for VR videos – including Google Jump [5, 6], and Facebook Surround360 [12] – have arrived at remarkably similar hardware: multi-view camera rigs with 14–16 cameras that are packed into a tight ring to capture omnidirectional stereo video. As of 2019, this defined the state of the art in widely available virtual reality imagery and video.

References

1. Ishiguro H, Yamamoto M, Tsuji S (1992) Omnidirectional stereo. IEEE Trans Pattern Anal Mach Intell 14(2):257–262. ISSN 0162-8828. https://doi.org/10.1109/34.121792
2. Gluckman J, Nayar SK, Thoresz KJ (1998) Real-time omnidirectional and panoramic stereo. In: Proceedings of the Image Understanding Workshop
3. Peleg S, Ben-Ezra M, Pritch Y (2001) Omnistereo: panoramic stereo imaging. IEEE Trans Pattern Anal Mach Intell 23(3):279–290. ISSN 0162-8828. https://doi.org/10.1109/34.910880
4. Richardt C, Pritch Y, Zimmer H, Sorkine-Hornung A (2013) Megastereo: constructing high-resolution stereo panoramas. In: Proceedings of the International Conference on Computer Vision and Pattern Recognition (CVPR), pp 1256–1263. https://doi.org/10.1109/CVPR.2013.166
5. Anderson R, Gallup D, Barron JT, Kontkanen J, Snavely N, Hernandez C, Agarwal S, Seitz SM Jump: virtual reality video. ACM Trans Graph (Proc SIGGRAPH Asia) 35(6):198:1–13 (2016) ISSN 0730-0301. https://doi.org/10.1145/2980179.2980257
6. Schroers C, Bazin J-C, Sorkine-Hornung A (2018) An omnistereoscopic video pipeline for capture and display of real-world VR. ACM Trans Graph 37(3):37:1–13. ISSN 0730-0301. https://doi.org/10.1145/3225150
7. Konrad R, Dansereau DG, Masood A, Wetzstein G (2017) SpinVR: towards live-streaming 3D virtual reality video. ACM Trans Graph (Proc SIGGRAPH Asia) 36(6):209:1–12. ISSN 0730-0301. https://doi.org/10.1145/3130800.3130836
8. Tanaka K, Tachi S (2005) TORNADO: omnistereo video imaging with rotating optics. IEEE Trans Vis Comput Graph 11(6):614–625 ISSN 1077-2626. https://doi.org/10.1109/TVCG.2005.107
9. Aggarwal R, Vohra A, Namboodiri AM (2016) Panoramic stereo videos with a single camera. In: Proceedings of the International Conference on Computer Vision and Pattern Recognition (CVPR), pp 3755–3763. https://doi.org/10.1109/CVPR.2016.408

10. Chapdelaine-Couture V, Roy S (2013) The omnipolar camera: a new approach to stereo immersive capture. In: Proceedings of the International Conference on Computational Photography (ICCP). https://doi.org/10.1109/ICCPhot.2013.6528311
11. Matzen K, Cohen MF, Evans B, Kopf J, Szeliski R (2017) Low-cost 360 stereo photography and video capture. ACM Trans Graph (Proc SIGGRAPH) 36 (4):148:1–12. ISSN 0730-0301. https://doi.org/10.1145/3072959.3073645
12. Facebook (2016) Surround 360 instruction manual. GitHub, July 2016. https://github.com/facebook/Surround360/blob/master/surround360_design/assembly_guide/Surround360_Manual.pdf

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, 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, 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

Omnistereo

▸ Omnidirectional Stereo

Opacity

Ivo Ihrke
Independent Scholar, Adelmannsfelden, Germany

Synonyms

Opaque

Related Concepts

▸ Transparency

Definition

Opacity is the property of objects to not transmit light. This non-transmittance can be due to several factors: (1) absorption, (2) scattering, and (3) reflection.

The corresponding adjective is "opaque." Opaque objects are neither transparent nor translucent. Sometimes the opposite property is referred to as *transmittance* (mostly in Physics). In Computer Graphics, the opposite of opacity is typically referred to as *transparency*.

The word opaque is often used in conjunction with absorbing objects; however, it is important to note that reflecting objects like mirrors are also opaque.

Background

A common notion is that 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. However, with slightly more precision, objects are required to be diffusely opaque: in particular, specularly reflective objects are opaque by the above 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* [5]. This technique uses a colored background in order to estimate the alpha channel.

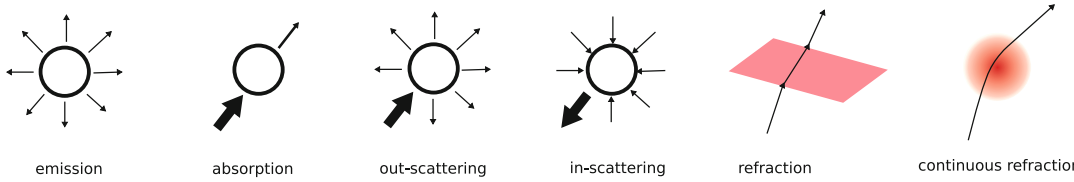
Theory

The following contains a discussion of the physical (optical) principles causing the perception of opacity.

The four main interactions of radiation with matter are absorption, scattering, reflection, and transmittance. Accordingly, the overall radiant energy, i.e., the energy carried by electromagnetic radiation, at a non-emissive point in space is, due to conservation of energy, given by [9]:

$$Q_{\mathrm{tot}}(\lambda) = Q_{\mathrm{a}}(\lambda) + Q_{\mathrm{s}}(\lambda) + Q_{\mathrm{r}}(\lambda) + Q_{\mathrm{t}}(\lambda). \quad (1)$$

Here, Q_{tot} is the total amount of energy, whereas Q_{a} is the absorbed, Q_{s} the scattered,



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

Q_r the reflected, Q_t the transmitted amount of energy, and λ the wavelength of light. Division by Q_{tot} results in a fractional formula:

$$1.0 = A + S + R + T, \tag{2}$$

where the terms, in order, correspond to the formula given above. Opaque objects are therefore those for which $T = 0$. An opaque object is neither transparent ($A = S = 0, R < 1$) nor translucent ($A + S + R < 1$).

In general, as indicated in Eq. 1, the values are wavelength dependent and therefore the source of color [9]. In some circumstances, the absorption and re-emission can change the wavelength (fluorescence/phosphorescence) and incur temporal delays (fluorescence $\approx 10^{-8}s$, phosphorescence up to several hours). A more refined version of Eq. 1 is therefore:

$$Q_{tot} = \int_{\Omega} \int_{\lambda_{min}}^{\lambda_{max}} \int_{t_0}^{\infty} Q_a(x, \lambda, t) + Q_s(x, \lambda, t) + Q_r(x, \lambda, t) + Q_t(x, \lambda, t) dt d\lambda dx. \tag{3}$$

As an example, consider a laser pulse with wavelength λ_0 hitting an object at position x_0 at time t_0. If $Q_{tot}(x_{BS}, \lambda, t) = 0$ for a suitably chosen set of points x_{BS} defining the backside of the object (Typically, the points x_{BS} are in the geometrical shadow of the object with respect to the light source.) for all $t \geq t_0$, $\lambda \in [\lambda_{min}, \lambda_{max}]$, the object is said to be opaque to wavelength λ_0.

In computer vision, we typically assume λ_0 to be in the visible part of the spectrum, i.e., between approximately 380 and 650 nm. The wavelength dependence is important since practically all materials are transparent in some wavelength range. As an example, common materials are transparent to x-rays, but materials that are transparent in the visible can be opaque in other wavelength ranges, e.g., optical glass in the UV range.

Mechanisms

A look at the three main mechanisms that may render a material opaque further elucidates the property opacity. In general, the opacity depends on the thickness of material that an object consists of which is the reason to attribute opacity to objects rather than to materials.

Absorption Absorption occurs due to mainly two mechanisms: atomic and molecular absorption and absorption in solids.

(a) atomic and molecular absorption When a photon's energy is absorbed by the electronic orbitals of an atom or molecule, atomic or molecular absorption occurs. In this case, the photon's energy $Q_e = h\nu$ must match the energy difference between two atomic/molecular energy levels, the lower one of which is occupied by an electron. Here, ν is the photon frequency and h is Planck's constant. The electron takes up the photon's energy and occupies the upper energy level after the interaction, leaving the atom/molecule in an excited state. This effect gives rise to discrete absorption lines as, for example, seen in the solar spectrum (Fraunhofer lines). The exact energy levels are subject to many physical conditions such as the vibrational and rotational state of the atom, the presence of magnetic fields, the presence of nearby other atoms or molecules, etc. These conditions lead to discretely split or continuously broadened energy levels. Their prediction is in the domain of quantum mechanics.

Atomic and molecular absorption will explain the absorption seen in gases and liquids.

(b) absorption in solids The tight arrangement of atoms in a solid has a strong influence on the absorption properties of a material. One generally distinguishes *metals* or *conductors*, *semi-conductors*, and *insulators* or *dielectrics*. Conduction takes place when free electrons are present in the material. This is the case if the material's band (Bands replace the atomic orbitals in dense substances since the individual atoms' orbitals merge. The band structure is temperature dependent.) gap is negligible or non-existent. Since light is electromagnetic radiation, it acts on charged objects like the free electrons, exerting a force, the so-called Lorentz force, on them. The force results in an acceleration of the electrons, increasing their velocities, which results in increased heat. Conduction leads to very efficient absorption, i.e., a transmitted wave is only entering conducting materials up to a depth of small fractions of a wavelength (on the order of the *skin depth* [3], II-32-7). This effect is, e.g., exploited in optical thin film sensors.

In semi-conductors, the electrons are lifted across the band gap, which must be smaller than the absorbed photon's energy. This photo-electric effect, discovered by Hertz and Hallwachs and explained by Einstein, is used in photo-detectors like CCDs and CMOS.

Absorption in insulators is mainly caused by the interaction of the electromagnetic wave with the electron orbitals and nuclei. The orbitals and nuclei are put into vibration by the changing electromagnetic field, and energy is dissipated as heat within the materials. For a detailed discussion, see [3], II-32-3.

Scattering 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. If waves are scattered by obstacles that are small in comparison to the wave length, the scattering is often termed *diffraction*. Properly speaking, reflection and refraction can also be considered as (very directed) scattering processes. However, the term scattering is most often employed to refer to microscopic light/material interactions. A formal description of energy transfer in scattering media, of which light transport is a special case, can be found in [8].

If a scattering process preserves energy, it is known as elastic scattering. The terminology stems from mechanics where a perfectly elastic collision is one that conserves kinetic energy and momentum. In this case, photons are effectively deviated. This may include the absorption and re-emission of photons, however, keeping their energy and thus the wavelength λ_0 constant.

If the photon energy changes in the process of absorption and re-emission, the scattering is known as inelastic. Examples are the already mentioned effects of fluorescence and phosphorescence.

Reflection The effect of reflection is a direct consequence of the electromagnetic nature of radiation. At material boundaries, where the electromagnetic material constants (They are not constant for different wavelengths of light.) (specific conductivity, magnetic permeability and electric permittivity) change abruptly, Maxwell's equations result in boundary conditions that necessitate a reflected and a transmitted part in response to an incident electromagnetic wave [1]. It is physically impossible to have only one of them. Therefore, reflection and transmission always occur when light interacts with a material surface. The reflected part may be minimized by anti-reflecting coatings. Similarly, the transmitted part may be absorbed very quickly, as in the case of metals (in the visible part of the spectrum).

Summary Opacity is a very broad term that covers a wide variety of different physical effects. Perfect opacity depends on the material thickness, the bulk material properties, possibly even the object shape, etc. It is a gradual process throughout the volume of an object (even of at

rather shallow depth beneath the surface) rather than a binary classification scheme. When the gradual effect is important, we speak of attenuation (see below).

Further Reading The classic textbook by Born and Wolf [1] and the informative book by Tilley [9] feature a detailed discussion of the physical effects pertaining to the concept of opacity. A classic intuitive explanation can be found in the Feynman Lectures on Physics [3], in particular lecture I-31 "The Origin of the Refractive Index," lecture II-32 "Refractive Index of Dense Materials," and lecture II-33 "Reflection from Surfaces."

Modeling

As outlined above, there is no 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 microscopic origins involved have been explained above. However, there are also macroscopic effects that may give rise to (partial) opacity.

An example is *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.

Correspondingly there is a wide variety of models of different complexity and applying to different scales of reality. Some of them are physical (to be outlined first), and some are purely phenomenological such as alpha blending in computer vision.

Attenuation The attenuation of light is due to the physical joint processes of absorption and scattering as outlined above. On the particle level, the absorption and scattering processes are statistical. Their net effect, however, is typically described in a continuous manner. In this description, the magnitude of absorption/scattering depends on the path length of light in the traversed medium. It is given by the *Beer-Lambert Law*:

$$\frac{L}{L_0} = \exp^{-\int_{s_0}^{s_1} \sigma_s \circ c(s) + \sigma_a \circ c(s) ds}. \quad (4)$$

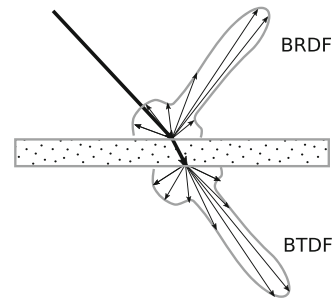
The ratio L/L_0 of outgoing over incident radiance is known as the *optical path length* in the interval $[s_0, s_1]$. Here, L_0 is the incident radiance and L is the radiance after attenuation. (Fresnel reflection (see below) is not included as a source of transmitted energy loss.) The path of the photon 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

$$\ln(L/L_0) = -\int_{s_0}^{s_1} \sigma_s \circ c(s) + \sigma_a \circ c(s) ds \quad (5)$$

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.



Scattering can occur both in transmission and in reflection. As mentioned, the effect is statistical

in nature. The prevalent modeling in the computer vision literature uses distribution functions that are essentially probabilities that a photon will be reflected/refracted or scattered in different directions.

We first discuss models for scattering at material interfaces. Reflection scattering is usually described by the (hemispherical) *bidirectional reflection distribution function* (BRDF), whereas transmissive scattering is described by the (hemispherical) *bidirectional transmission distribution function* (BTDF). These functions describe scattering at an interface or due to thin transmissive layers of material. Sometimes, the combined effect of reflection and transmission at an interface is described by a single (spherical) *bidirectional scattering distribution function* (BSDF).

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. If only a single scattering event occurs, the process is called *single scattering*; otherwise it is referred to as *multiple scattering*. If both scattering and absorption occur, in-scattering can have an effect on the attenuation coefficient described above.

Standard physical models for scattering include *Rayleigh scattering* for scattering at microscopic particles much smaller than the wavelength of light and *Mie scattering* for scattering off spherical particles. For a detailed discussion, see e.g., [4].

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, i.e., the ray changes direction upon being transmitted into the material. 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, and
- dynamic scene, dynamic background, i.e., video matting [2].

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

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 [7] 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, even dynamically, have shown good success.

References

1. Born M, Wolf E (1999) Principles of Optics, 7th edn. England, Cambridge University Press
2. Chuang Y-Y, 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, pp 243–248, New York. ACM
3. Feynman RP, Leighton RB, Sands ML (1964–1966) The Feynman lectures on physics, 3 vols. Library of Congress Catalog Card No. 63-20717
4. Frezza F, Mangini F, Tedeschi N (2018) Introduction to electromagnetic scattering: tutorial. J Opt Soc Am A 35(1):163–173
5. Millerson JOG (2009) Television Production. Focal Press, New York & Philadelphia
6. Ihrke I, Magnor M (2004) Image-based tomographic reconstruction of flames. In: ACM Siggraph/Eurographics Symposium Proceedings, Symposium on Computer Animation, pp 367–375
7. 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, pp 427–437, New York. ACM
8. Siegel R, Howell JR (1992) Thermal radiation heat transfer, 3rd edn. Taylor & Francis, New York
9. Tilley RJD (2010) Colour and the optical properties of materials. New Jersey, John Wiley & Sons

Opaque

▸ Opacity

Optic Flow

▸ Optical Flow: Traditional Approaches

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, San Diego, pp 53–84

Optical Center

▸ Center of Projection

Optical Flow: Traditional Approaches

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 (3) by nonquadratic ones [3, 4]. This can be interpreted as replacing the inherent Gaussian distribution model in (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 (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 post-processing 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

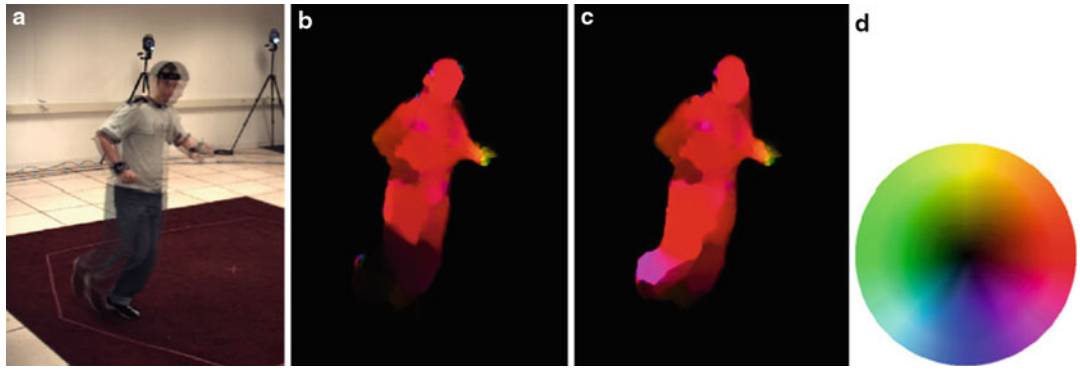
Typical applications of optical flow are tracking, motion segmentation, and action recognition. Tracking has again many applications in fields like structure-from-motion and surveillance. The most widely spread tracker, the KLT tracker [15], is based on the Lucas-Kanade technique described above. Trackers 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



Optical Flow: Traditional Approaches, 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

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 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) Comple-

mentary 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. Ph.D. thesis, Faculty of Mathematics and Computer Science, Saarland University
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, Okayama University, Okayama, Japan

Synonyms

Optimal parameter estimation

Related Concepts

- ▸ Ellipse Fitting
- ▸ Fundamental Matrix
- ▸ Maximum Likelihood Estimation

Definition

Optimal estimation in the computer vision context 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 2D/3D 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*

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

with respect to the true values $\bar{\boldsymbol{x}}_\alpha$ subject to given knowledge about them. Hereafter, $\langle \boldsymbol{a}, \boldsymbol{b}\rangle$ 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 identity 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 $\bar{\boldsymbol{x}}_\alpha$. In this case, I is often referred to as the *reprojection error*. That name originates from the following intuition: In inferring the 3D structure of the scene from its projected images, maximum likelihood under homogeneous and isotropic Gaussian noise means *reprojecting* the inferred 3D 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

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

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

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

The unknown parameter $\boldsymbol{\theta}$ allows one to infer the 2D/3D 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." This model is called *structural*. The 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 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; we

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, 2]. 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 the Mahalanobis distance (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 [3]. 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, 2].

According to the experience of many vision researchers, maximum likelihood is known to produce highly accurate solutions. A major concern is its computational burden, because maximum likelihood usually requires complicated nonlinear optimization. The standard approach is to express each of $\bar{\boldsymbol{x}}_\alpha$ explicitly in terms of $\boldsymbol{\theta}$ by introducing some auxiliary parameters, or nuisance parameters. After all the expressions are substituted back into (1), the Mahalanobis distance I becomes a function of $\boldsymbol{\theta}$ and the nuisance 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

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

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 2D point sequence or a parametric surface such as a plane, an ellipsoid, and a polynomial surface to a noisy 3D point sets and computing the fundamental matrix or the homography from noisy point correspondences over two images [4, 5]. 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

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

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

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

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 of $\boldsymbol{V}_\alpha$, which is equal to the number of independent equations of (4). The name Sampson error stems from the classical ellipse fitting scheme [6].

The Sampson error (7) can be minimized by various means including the *FNS* (*Fundamental Numerical Scheme*) [7] and the *HEIV* (*Heteroscedastic Errors-in-Variable*) [8]. 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 [5, 9]. 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

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

where $\bar{\boldsymbol{\xi}}_\alpha^{(k)}$ are the true values 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* [1,2,5]. 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, 2].

Algebraic Methods

Recently, there has been a remarkable progress in the study of algebraic methods. By "algebraic methods," we mean we solve some "algebraic equations" (directly or iteratively), rather than minimizing some cost function such as the reprojection error. Originally, algebraic methods were thought of as an auxiliary to maximum likelihood and used for initialization of maximum likelihood iterations. In the last decade, however, it has been found that some algebraic methods outperform maximum likelihood in accuracy [5].

Algebraic methods solve a nonlinear equation in the form

$$\boldsymbol{M}\boldsymbol{\theta} = \lambda\boldsymbol{N}\boldsymbol{\theta}, \tag{10}$$

with

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

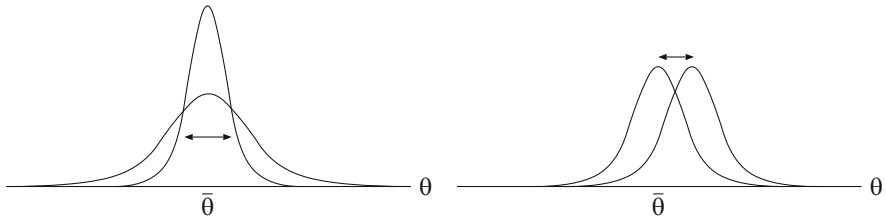
where $W_\alpha^{(kl)}$ are some weights that depend on $\boldsymbol{\theta}$. Various methods with different names arise according to the choice of the weights $W_\alpha^{(kl)}$ and the matrix $\boldsymbol{N}$ in (10). This scheme was originally motivated to minimize $\langle\boldsymbol{\theta}, \boldsymbol{M}\boldsymbol{\theta}\rangle$, which is called the (weighted) *algebraic distance*, hence the name "algebraic method," subject to the con-

straint $\langle \boldsymbol{\theta}, \boldsymbol{N}\boldsymbol{\theta} \rangle = \text{constant}$. The solution of (10) is obtained by iteration: we first regard $W_\alpha^{(kl)}$ and $\boldsymbol{N}$ as given and solve the generalized eigenvalue problem (10), then update $W_\alpha^{(kl)}$ and $\boldsymbol{N}$ using the resulting $\boldsymbol{\theta}$, and repeat this process.

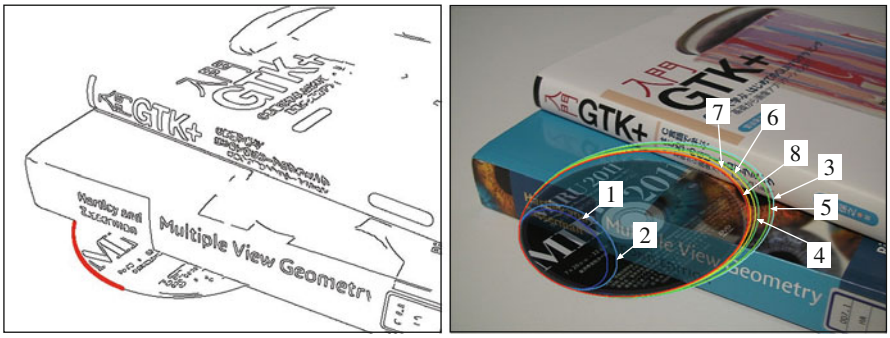
If we choose $W_\alpha^{(kl)} = 1$ and $\boldsymbol{N} = \boldsymbol{I}$ (hence no iterations are necessary), this method is nothing but the standard *least squares*. If we choose $W_\alpha^{(kl)} = 1$ and $\boldsymbol{N} = V_0[\boldsymbol{x}_\alpha]$ (the covariance matrix of $\boldsymbol{x}_\alpha$ up to scale), this reduces the method of Taubin [10], which is known to produce a fairy accurate solution. Efforts were made to improve the accuracy of the Taubin method by choosing optimal $\boldsymbol{N}$, resulting in a scheme called *HyperLS* [11], which is non-iterative. If we use the same $\boldsymbol{N}$ as in the Taubin method but use the weights $W_\alpha^{(kl)}$ that appear in (7) and (8), we obtain the iterative scheme of *renormalization* [12]. Like for HyperLS, we can optimize the matrix $\boldsymbol{N}$ of renormalization to obtain *hyper-renormalization* [13], which exhibits higher accuracy than maximum likelihood [5].

The superiority of hyper-renormalization is confirmed by statistical analysis. If we regard the input $\boldsymbol{x}_\alpha$ as random variables, the computed solution $\boldsymbol{\theta}$ of (10) is also a random variable. It can be shown that the matrices $\boldsymbol{M}$ and $\boldsymbol{N}$ of (10) control, respectively, the covariance and the bias of $\boldsymbol{\theta}$ (Fig. 1) and that the matrices $\boldsymbol{M}$ and $\boldsymbol{N}$ of hyper-renormalization are such that the covariance of $\boldsymbol{\theta}$ reaches the KCR lower bound up to $O(\sigma^4)$ and the bias of $\boldsymbol{\theta}$ is $\boldsymbol{0}$ up to $O(\sigma^4)$.

On the other hand, efforts have been made to improve the accuracy of maximum likelihood by correcting the solution a posteriori, called *hyperaccurate correction* [14]. It can be shown that maximum likelihood followed by hyperaccurate correction can achieve equivalent accuracy to hyper-renormalization [5] (Fig. 2). However, iterations for computing maximum likelihood solution sometimes fail to converge in the presence of large noise, compared to which hyper-renormalization iterations are rather robust.



Optimal Estimation, Fig. 1 *Left*: The matrix $\boldsymbol{M}$ controls the covariance of $\boldsymbol{\theta}$. *Right*: The matrix $\boldsymbol{N}$ controls the bias of $\boldsymbol{\theta}$



Optimal Estimation, Fig. 2 *Left*: An edge image of a scene with a circular object. An ellipse is fitted to the 160 edge points indicated. *Right*: Fitted ellipses superimposed on the original image. The occluded part is artificially composed for visual ease. (1) Least squares, (2) iterative reweight, (3) Taubin method, (4) renormalization, (5) HyperLS, (6) hyper-renormalization, (7) ML, (8) ML followed by hyperaccurate correction. (From [4])

References

1. Chernov N, Lesort C (2004) Statistical efficiency of curve fitting algorithms. Comput Stat Data Anal 47(4):713–728
2. Kanatani K (2008) Statistical optimization for geometric fitting: theoretical accuracy analysis and high order error analysis. Int J Comput Vis 80(2):167–188
3. Neyman J, Scott EL (1948) Consistent estimates based on partially consistent observations. Econometrica 16(1):1–32
4. Kanatani K, Sugaya Y, Kanazawa Y (2016a) Ellipse fitting for computer vision: implementation and applications. Morgan Claypool, San Rafael
5. Kanatani K, Sugaya Y, Kanazawa Y (2016b) Guide to 3D vision computation: geometric analysis and implementation. Springer, Cham
6. Sampson PD (1982) Fitting conic sections to "very scattered" data: an iterative refinement of the Bookstein algorithm. Comput Graphics Image Process 18(1):97–108
7. Chojnacki W, Brooks MJ, van den Hengel A, Gawley D (2000) On the fitting of surfaces to data with covariances. IEEE Trans Patt Anal Mach Intell 22(11):1294–1303
8. Leedan Y, Meer P (2000) Heteroscedastic regression in computer vision: problems with bilinear constraint. Int J Comput Vis 37(2):127–150
9. Kanatani K, Sugaya Y (2010) Unified computation of strict maximum likelihood for geometric fitting. J Math Imaging Vis 38(1):1–13
10. 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 Patt Anal Mach Intell 13(11):1115–1138
11. Kanatani K, Rangarajan P, Sugaya Y, Niitsuma H (2011) HyperLS for parameter estimation in geometric fitting. IPSJ Trans Comput Vis Appl 3:80–94
12. Kanatani K (1993) Renormalization for unbiased estimation. In: Proceedings of 4th international conference on computer vision, Berlin, pp 599–606
13. Kanatani K, Al-Sharadqah A, Chernov N, Sugaya Y (2014) Hyper-renormalization: non-minimization approach for geometric estimation. IPSJ Trans Comput Vis Appl 6:143–149
14. Kanatani K, Sugaya Y (2013) Hyperaccurate correction of maximum likelihood for geometric estimation. IPSJ Trans Comp Vis Appl 5:19–29

Optimal Parameter Estimation

▸ Optimal Estimation

Oren-Nayar Reflectance Model

Ping Tan
School of Computing Science, Simon Fraser University, Vancouver, BC, Canada

Related Concepts

▸ Bidirectional Reflectance Distribution Function
▸ Diffuse Reflectance
▸ Radiance
▸ Reflectance Models

Definition

The Oren-Nayar model computes 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 Lambert's model [2] which assumes the light is uniformly reflected in all 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 accounts for 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 accounts for the light that enters the surface and is distributed more uniformly in all directions.

In computer vision, 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].

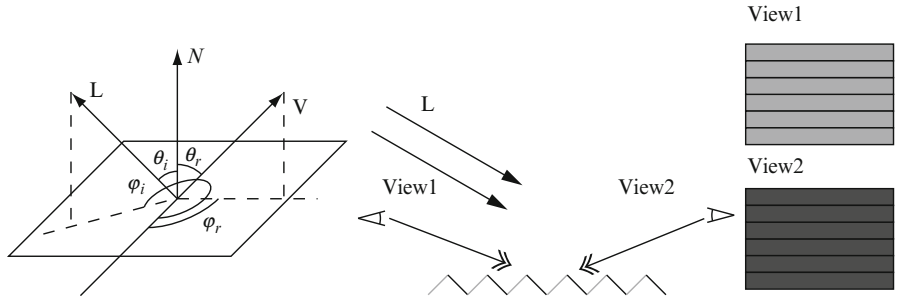
Backscattering can be well explained by studying the surface microstructures. As shown on the right of Fig. 1, when the polar angle of the incident light θ_i is large, some of the microstructures are in shadows. When the surface is viewed from a direction $View1$ that is similar to the incident direction L, all visible microstructures are illuminated and the surface appears brighter. However, when the surface is viewed from the direction $View2$, 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]. Firstly, a reflectance model $I^{(1)}$ is derived for anisotropic surfaces consisting of V-cavities of the same slope and the same direction. This model includes two components, one for direct illumination and the other for interreflection between microfacets. Secondly, a reflectance model $I^{(2)}$ is derived for isotropic surfaces consisting of V-cavities with the same slope but all kinds of directions in a plane. This model $I^{(2)}$ is derived by integrating $I^{(1)}$ over different cavity directions. Thirdly, the final Oren-Nayar model is derived for general isotropic surfaces that consist of V-cavities whose slopes 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 includes a component $I_1^{(3)}$ for direct illumination and a component $I_2^{(3)}$ for microfacet interreflection. These two components are defined as the following, respectively,

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

and

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



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

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 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 captures the *backscattering* effect. An interesting fact about this model is that it degenerates to 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 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:

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

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.

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:

$$\frac{x^2 + y^2}{2}, \quad (2)$$

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

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

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:

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

with:

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

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

$$\frac{1}{2}\left(k_1u^2 + k_2v^2\right), \text{with } k_1 \geq k_2. \tag{6}$$

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:

$$a_{20} = \frac{k_1 + k_2}{2} + \frac{k_1 - k_2}{2}\cos\ 2\varphi, \tag{7}$$

$$a_{11} = \frac{k_1 - k_2}{2}\sin\ 2\varphi, \tag{8}$$

$$a_{02} = \frac{k_1 + k_2}{2} - \frac{k_1 - k_2}{2}\cos\ 2\varphi, \tag{9}$$

and, equivalently:

$$r = \frac{k_1 - k_2}{2}\cos\ 2\varphi, \tag{10}$$

$$s = \frac{k_1 - k_2}{2}\sin\ 2\varphi, \tag{11}$$

$$t = \frac{k_1 + k_2}{2}. \tag{12}$$

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

$$k = \sqrt{\frac{k_1^2 + k_2^2}{2}}, \tag{13}$$

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

$$\sigma = \arctan\frac{k_1 + k_2}{k_1 - k_2}. \tag{14}$$

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

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

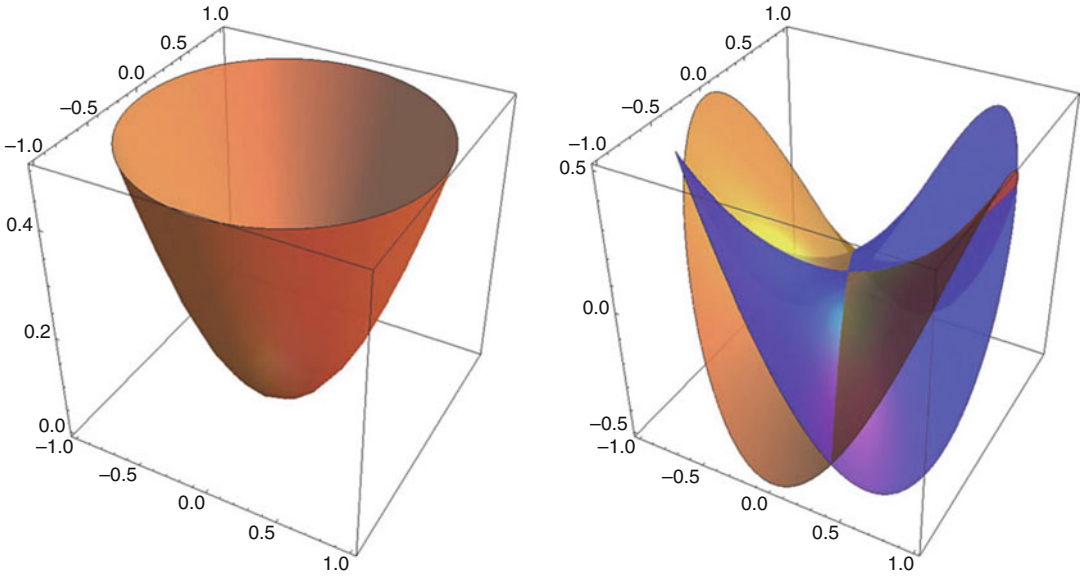
and

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

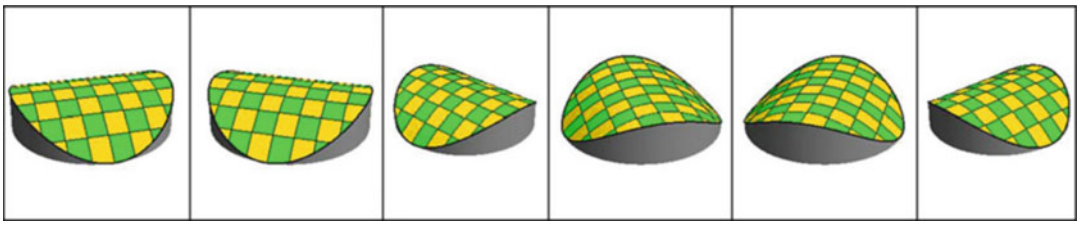
This can be further simplified by identifying the mean curvature $2H = \kappa_1 + \kappa_2 = a_{20} + a_{02}$, the Gaussian curvature $K = k_1k_2 = a_{20}a_{02} - a_{11}^2$, and the "bending energy" $E = k_1^2 + k_2^2 = a_{20}^2 + 2a_{11}^2 + a_{02}^2$. (Notice that H and K should be distinguished from 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:

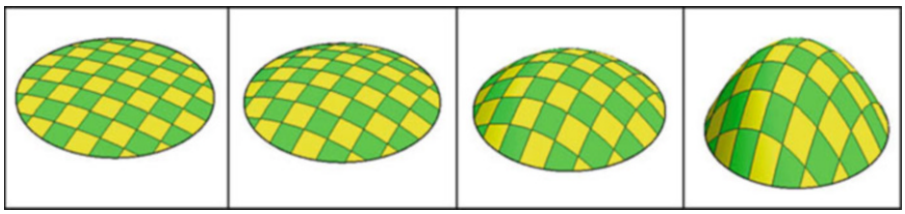
$$k = \sqrt{\frac{k_1^2 + k_2^2}{2}}, \tag{17}$$



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



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

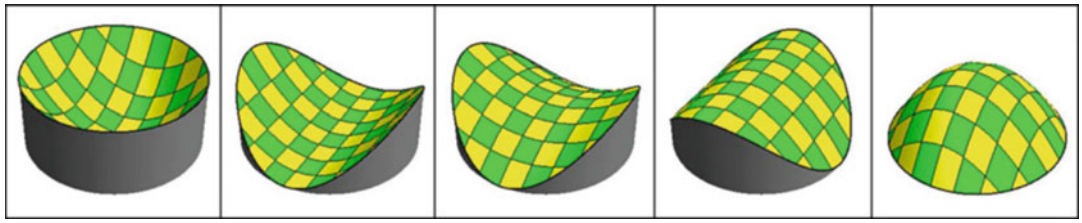


Osculating Paraboloids, Fig. 3 A scale of surfaces of the same shape, the Casorati curvature varying by factors of two

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:

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

$$\text{and} \quad [F]_\mu = \sqrt{\langle F^2\rangle_\mu - \langle F\rangle_\mu^2}, \tag{18}$$



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 $-\pi/2$

one has:

$$k = \frac{\left[k_1 x^2 + k_2 y^2\right]}{\left[x^2 + y^2\right]} \tag{19}$$

(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:

$$\varrho = \sqrt{r^2 + s^2 + t^2} = k \tag{20}$$

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

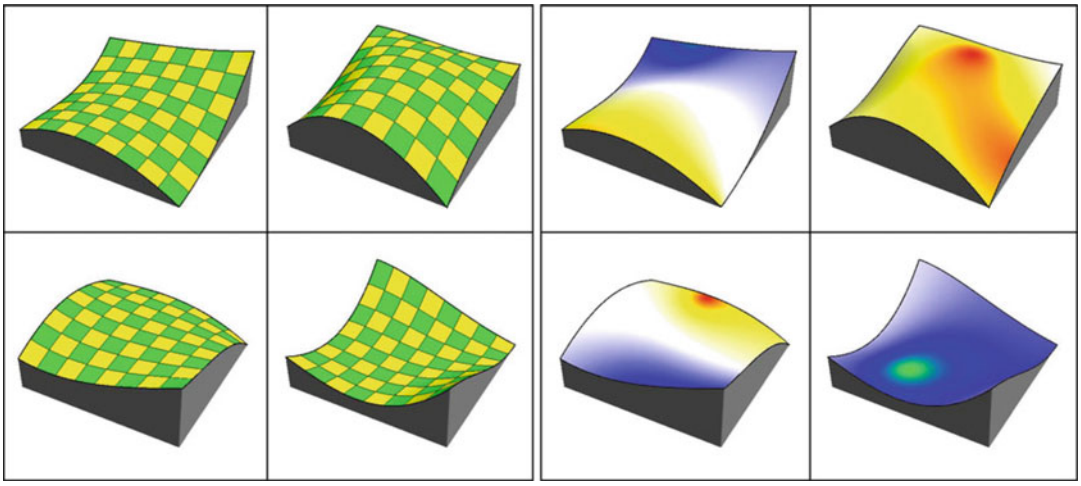
$$\phi = \arctan \frac{s}{r} = 2\varphi. \tag{22}$$

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, 6, 7, and 8).

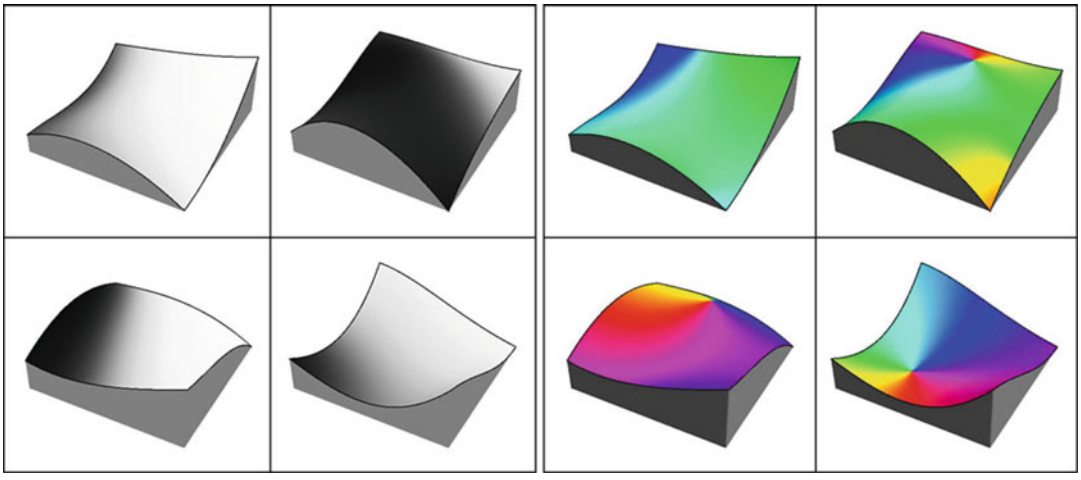
On a general curved surface the shape of the local osculating paraboloids will change from point to point (see Figs. 5, 6, and 7). Since the change will be smooth, the surface can be mapped on a surface in shape space, a surface that may well be expected to have self-intersections and perhaps not everywhere 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:



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*



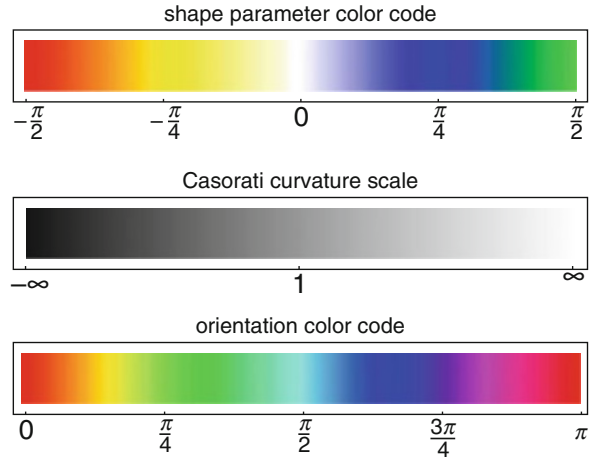
Osculating Paraboloids, Fig. 6 At *left* the surfaces of Fig. 5 left have been shaded with the Casorati curvature. The *gray scale* represents a nonlinear monotonic function of the logarithm of the curvature. At right the surfaces have been colored with the orientation of the direction of largest principal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

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

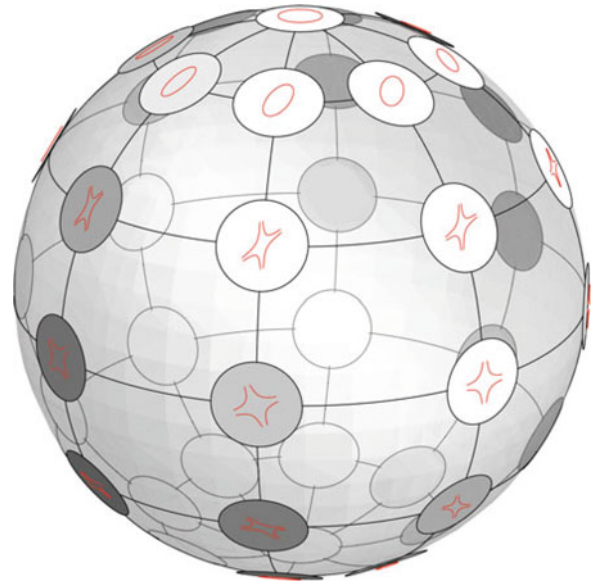
In an infinitesimal neighborhood of the origin one finds:

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

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



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

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

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

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

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

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 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:

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

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