EUROPEAN PATENT SPECIFICATION

LEVERAGING MULTI CUES FOR FINE-GRAINED OBJECT CLASSIFICATION

NUTZUNG VON MEHREREN HINWEISEN FÜR FEINKÖRNIGE OBJEKTKLASSIFIZIERUNG

UTILISATION DE REPÈRES MULTIPLES POUR UN CLASSEMENT D'OBJETS À GRAINS FINS

Note: Within nine months of the publication of the mention of the grant of the European patent in the European Patent Bulletin, any person may give notice to the European Patent Office of opposition to that patent, in accordance with the Implementing Regulations. Notice of opposition shall not be deemed to have been filed until the opposition fee has been paid. (Art. 99(1) European Patent Convention).
Description

[Technical Field]

[0001] One or more embodiments relate generally to object classification, and in particular, a system and a method and a storage medium for leveraging multiple cues for fine-grained object classification.

[Background Art]

[0002] Over the years, software application designers employ fine-grained object classification more frequently than generic object classification during software application development. Fine-grained object classification involves at least the following steps: (1) labeling/localizing discriminative parts, and (2) learning appearance descriptors. Conventionally, labeling/localizing discriminative parts is performed using either manual parts annotation or image segmentation, and learning appearance descriptors is performed using a multi-layer deep neural network. As objects classed in a fine-grained object category (i.e., class) share a high degree of shape similarity, performing the steps of labeling/localizing discriminative parts and learning appearance descriptors may be challenging.

[0003] US 8,837,779 B2 describes a method for determining the pose of a camera relative to a real environment comprises the following steps: taking at least one image of a real environment by means of a camera, the image containing at least part of a real object, performing a tracking method that evaluates information with respect to correspondences between features associated with the real object and corresponding features of the real object as it is contained in the image of the real environment, so as to obtain conclusions about the pose of the camera, determining at least one parameter of an environmental situation, and performing the tracking method in accordance with the at least one parameter. Analogously, the method can also be utilized in a method for recognizing an object of a real environment in an image taken by a camera.

[Disclosure]

[Technical Solution]

[0004] The invention is as defined in the independent claims. Preferred embodiments are defined in the dependent claims.

[Advantageous Effects]

[0005] One or more embodiments provide a system and a method and a storage medium for leveraging multiple cues for fine-grained object classification.

[Description of Drawings]

[0006] These and other features, aspects and advantages of the one or more embodiments will become understood with reference to the following description and accompanying figures.

FIG. 1 illustrates an example electronic device including a fine-grained object classification framework, in one or more embodiments;
FIG. 2 illustrates an example training system, in one or more embodiments;
FIG. 3 illustrates an example object recognition system in detail, in one or more embodiments;
FIG. 4 illustrates an example process for building a feature database, in one or more embodiments;
FIG. 5 illustrates an example image alignment, in one or more embodiments;
FIG. 6 illustrates an example learned filter bank, in one or more embodiments;
FIG. 7 illustrates an example single-layer feature extraction scheme, in one or more embodiments;
FIG. 8 is a flowchart of an example process for fine-grained object recognition, in one or more embodiments; and
FIG. 9 is an exemplary high-level block diagram showing an information processing system comprising a computer system useful for implementing disclosed embodiments.

[Mode for Invention]

[0007] The invention is defined by the appended claims.
[0008] For expository purposes, the term "fine-grained object classification" as used herein generally refers to a type
of object classification that involves at least the following steps: (1) labeling/localizing discriminative parts, and (2) learning appearance descriptors.

[0009] The terms "object recognition" and "object classification" may be used interchangeably in this specification.

[0010] One or more embodiments relate generally to object classification, and in particular, a system and a method for leveraging multiple cues for fine-grained object classification. One embodiment provides a method comprising estimating a camera pose of an input image and aligning the input image to a desired camera pose based on a feature database. The input image comprises an image of an object. For example, the object is fine-grained object. The method further comprises classifying the object based on the alignment.

[0011] One embodiment provides a fine-grained object classification framework that utilizes three-dimensional (3D) camera pose estimation to align images and label/localize discriminative parts. The framework further utilizes a classification scheme including unsupervised feature extraction and classification. Unsupervised feature extraction may be performed utilizing different methods. Unsupervised feature extraction is a lightweight feature extraction scheme that facilitates learning appearance descriptors based on one or more constraints (e.g., constraints of a mobile device, such as small training data size, memory/computational costs, etc.). In one embodiment, unsupervised feature extraction is performed utilizing Convolutional Sparse Coding (CSC) and manifold learning. Similar to using a deep neural network, unsupervised feature extraction produces highly diverse feature filters, allowing capture of both low and high level features. Further, unlike a complex learning framework that utilizes a deep neural network, unsupervised feature extraction significantly reduces model complexity and preserve hierarchical level features. Also, unsupervised feature extraction may utilize a single-layer feature extraction scheme instead of a multi-layer deep neural network to reduce computation time.

[0012] FIG. 1 illustrates an example electronic device 100 including a fine-grained object classification framework, in one or more embodiments. The device 100 comprises one or more processor units 110 and one or more storage units 120. The one or more storage units 120 may maintain one or more databases 130. One or more applications may execute/operate on the processor units 110. In one embodiment, the applications include an object recognition system 190 resides on the device 100.

[0013] A user 30 may carry on his/her person the device 100. Examples of the device 100 include, but are not limited to, one of the following mobile devices: a tablet, a smart phone, a laptop, a wearable device, etc.

[0014] In one embodiment, the user 30 may utilize the device 100 to assist him/her in fine-grained object classification.

For example, during a mobile transaction, the user 30 may utilize the device 30 for point of sale (POS) terminal classification (i.e., determine a model of a POS terminal used for the mobile transaction). In one embodiment, the device 100 is configured to exchange data with an object, such as a POS terminal 50, over a connection (e.g., a WiFi connection or a cellular data connection).

[0015] In one embodiment, the device 100 is configured to exchange data with one or more remote servers 60 over a connection (e.g., a wireless connection such as a WiFi connection or a cellular data connection, a wired connection, or a combination of the two).

[0016] The device 100 further comprises one or more other input/output (I/O) modules 140, such as a keyboard, a keypad, a touch interface, a display screen, etc. The device 100 further comprises at least one sensor module 150 (e.g., a camera, a microphone, etc.) for capturing contextual and/or environmental data. For example, for POS terminal classification, a camera of the device 100 may be used to capture a photo of a POS terminal used during a mobile transaction.

[0017] In one embodiment, one or more learned models utilized by the object recognition system 200 for fine-grained object classification are trained offline (i.e., not on the device 100). For example, the learned models may be trained on a remote server 210 utilizing computation resources of the server 210 (e.g., one or more processors 601 and/or one or more storage devices 604). After training, the learned models may be loaded onto/downloaded to the device 100 as a portion of the object recognition system 200. In another embodiment, the learned models are trained on the device 100 itself.

[0018] FIG. 2 illustrates an example training system 190, in one or more embodiments. In one embodiment, in a training phase, the training system 190 trains one or more learned models utilized by the object recognition system 200 for fine-grained object classification (e.g., feature databases, filter banks, classification model, a projection matrix, etc.). In one embodiment, the training system 190 resides on a remote server 210. In another embodiment, the training system 190 resides on the device 100.

[0019] In one embodiment, the training system 190 comprises, but is not limited to, one or more of the following components: (1) a database builder 210 configured to build one or more two-dimensional (2D) to three-dimensional (3D) feature databases 320 (FIG. 3) for one or more fine-grained object categories (i.e., classes), and (2) a feature learning unit 234 configured to learn one or more 2D convolutional filter banks 197, a projection matrix 195 for reducing feature dimensions, and a classification model 196.

[0020] The training system 190 may maintain one or more databases. In one embodiment, the databases comprise one or more image databases 310 for one or more fine-grained object categories (e.g., CATEGORY1, ..., CATEGORYN).
Each image database 310 corresponds to a fine-grained object category. For each fine-grained object category, a corresponding image database 310 comprises a set 311 of training images 312 from the same fine-grained object category. In one embodiment, a set 311 of training images 312 includes a plurality of different 2D images 312 capturing different illumination changes, different views, and different backgrounds of a 3D object.

[0021] In the training phase, the database builder 210 builds, for each fine-grained object category, a corresponding feature database 320. Examples of different fine-grained object categories include, but are not limited to, different categories of POS terminals 50 (e.g., Verifone MX915, Verifone MX925, etc.).

[0022] In one embodiment, the database builder 210 comprises at least one of the following: (1) a structure from motion (SfM) unit 211 configured to apply a structure from motion approach for 3D reconstruction, and (2) a descriptors units 212 configured to add local 2D feature descriptors 332 to 3D points 322.

[0023] For a fine-grained object, a corresponding feature database 320 comprises a set 321 (FIG. 3) of sparse 3D points 322 (FIG. 3) representing sparse geometry of a shape of a 3D object. As described in detail later herein, in the training phase, the database builder 210 determines the set 321 of sparse 3D points based on a set 311 (FIG. 3) of training images 312 (FIG. 3) from the same fine-grained object category.

[0024] In the training phase, the database builder 210 is further configured to add, for each 3D point 322, a corresponding set 331 (FIG. 3) of local 2D feature descriptors 332 (FIG. 3) to the 3D point 322. Each 3D point 322 of the set 321 is associated with a corresponding set 331 of local 2D feature descriptors 332 indicative of a visual appearance of the 3D object surrounding the 3D point 322.

[0025] As described in detail later herein, in one embodiment, in the training phase, the feature learning unit 234 applies CSC to learn one or more 2D convolutional filter banks 197.

[0026] As described in detail later herein, in one embodiment, in the training phase, the feature learning unit 234 applies a manifold learning algorithm to learn a projection matrix for reducing feature dimensions (i.e., transforming original features to a low-dimensional space). The manifold learning algorithm applied may be Orthogonal Locality Preserving Projections (OLPP) or another manifold learning scheme.

[0027] In one embodiment, in the training phase, the feature learning unit 234 trains a learned classification model 196 for mapping a feature of a low-dimensional space to a corresponding fine-grained object category by training one or more support vector machines (SVMs) based on feature vectors. To prevent over-fitting, a validation set is randomly selected to optimize parameters for the SVMs. Based on cross-validation, there are many different experimental folds, and a set of parameters with best performance is selected.

[0028] FIG. 3 illustrates an example object recognition system 200 in detail, in one or more embodiments. The system 200 comprises, but is not limited to, one or more of the following components: (1) an image alignment unit 220, and (2) a fine-grained object classification unit 230.

[0029] In a deployment phase, the image alignment unit 220 is configured to: (1) in response to receiving an input image 313 (e.g., an image captured by a sensor module 150), estimate a camera pose of the input image 313, and (2) align the input image 313 to a desired camera pose using a learned feature database 320 (e.g., a learned feature database 320 trained by the training system 190 in the training phase).

[0030] In one embodiment, the image alignment unit 220 comprises at least one of the following: (1) a localization unit 221 configured to estimate camera information for an input image 313 (e.g., an intrinsic matrix, a rotation matrix, and a translation matrix) and a camera pose of the input image 313, and (2) an alignment unit 222 configured to produce an aligned image 360, wherein the aligned image 360 comprises a portion of the input image 313 aligned to a desired camera pose.

[0031] In the deployment phase, the fine-grained object classification unit 230 is configured to classify a fine-grained object included in the input image 313 based on the aligned image 360 received from the alignment unit 222 and a learned classification model (e.g., a learned classification model 196 trained by the training system 190 in the training phase). In one embodiment, the fine-grained object classification unit 230 comprises at least one of the following: (1) a feature extraction unit 231 configured to apply a single-layer feature extraction scheme that provides both low-level feature representation and high-level feature representation of the object, (2) a manifold learning unit 232 configured to apply a learned projection matrix (e.g., a learned projection matrix 195 trained by the training system 190 in the training phase) to reduce feature dimensions, and (3) a classifier 233 configured to map a feature to a corresponding fine-grained object category based on the learned classification model.

[0032] As stated above, the one or more storage units 120 may maintain one or more databases 130. In one embodiment, the databases 130 comprise one or more learned feature databases 320 (e.g., a learned feature database 320 trained by the training system 190 in the training phase) for one or more fine-grained object categories (e.g., CATEGORY1, ..., CATEGORYN). Each feature database 320 corresponds to a fine-grained object category. Each feature database 320 comprises a set 321 of sparse 3D points 322.

[0033] In one embodiment, the databases 130 further comprise one or more descriptors databases 330 (e.g., sets of descriptors 332 determined by the training system 190 in the training phase) for one or more 3D points 322 (e.g., 3D POINT1, ..., 3D POINTN). Each descriptors database 330 corresponds to a 3D point 322. Each descriptors database
EP 3 417 425 B1

330 comprises a set 331 of local 2D feature descriptors.

[0034] FIG. 4 illustrates an example process for building a feature database 320, in one or more embodiments. Let \( \{pi\} \) denote a set of sparse 3D points representing sparse geometry of a shape of a 3D object. A feature database 320 comprises a set of sparse 3D points \( \{p_i\} \). Let \( \{l_i\} \) denote a set of \( n \) training images 312, wherein \( n \geq 2 \). For example, if the 3D object is a POS terminal 50, the set of training images \( \{l_i\} \) may include a plurality of different 2D images 312A-312D capturing the POS terminal 50 at different angles. Let \( j \) denote a track representing a projection of a 3D point \( p_i \) on multiple training images 312 of a set of training images \( \{l_i\} \). Let \( \{d_{ij}\} \) denote a set of local 2D feature descriptors. Let \( \{C_j\} \) denote a set of unknown camera parameters.

[0035] In the training phase, for each fine-grained object category, the database builder 210 is configured to compute a set of oriented FAST, rotated BRIEF (ORB) 2D keypoints and descriptors based on each training image 312 of a set of training images \( \{l_i\} \) from the same fine-grained object category. Similar 2D features (i.e., 2D keypoints and descriptors) are matched and matching 2D features are organized across the set of training images \( \{l_i\} \) into a plurality of tracks. Each track \( j \) comprises a set of consistently matching 2D keypoints and descriptors across multiple training images 312 of the set of training images \( \{l_i\} \).

[0036] For a feature database 320 corresponding to a fine-grained object category, the database builder 210 determines a set of sparse 3D points \( \{p_i\} \) to include in the feature database 320 based on a set of training images \( \{l_i\} \) from the same fine-grained object category. In one embodiment, positions of the set of sparse 3D points \( \{p_i\} \) are computed based on bundle adjustment. Let \( K_i \) denote an intrinsic matrix, let \( R_i \) denote a rotation matrix, and let \( T_i \) denote a translation matrix. The matrices \( K_i, R_i, \) and \( T_i \) may be constructed based on a set of unknown camera parameters \( \{C_i\} \). Let \( P \) denote a projection equation from a 3D point \( p_j \) to a 2D keypoint \( k \) on a training image \( l_i \) of the set \( \{l_i\} \). In one embodiment, the projection equation \( P \) may be represented in accordance with equation (1) provided below:

\[
k = P(C_i, p_j) = K_i[R_i][T_i][p_j]
\]  

(1)

wherein \( C_i \) denotes an unknown camera parameter of the set \( \{C_i\} \).

[0037] In one embodiment, one or more re-projection errors are minimized in accordance with equation (2) provided below:

\[
\sum_i \sum_j \|k_{ij} - P(C_i, p_j)\|
\]  

(2)

wherein \( k_{ij} \) denotes a 2D keypoint from the training image \( l_i \) in a track \( j \) representing a projection of the 3D point \( p_j \). The unknown camera parameter \( C_i \) and the 3D point \( p_j \) may be computed using equation (2) provided above.

[0038] Let \( J \) denote a Jacobian matrix for the projection equation \( P \), let \( H \) denote a Hessian approximation, and let \( d^1 \) denote a damping matrix, wherein \( d \) is a scalar, and \( I \) is an identity matrix. In one embodiment, to estimate a camera pose, a Jacobian matrix \( J \) for the projection equation \( P \) is computed, and a Hessian approximation \( H \) is assembled using the Jacobian matrix \( J \). A damping matrix \( d^1 \) is applied to the Hessian approximation \( H \). If an estimated camera pose of an image causes a large re-projection error, the scalar \( d \) may be adjusted to a larger value in a subsequent iteration; otherwise, the scalar \( d \) may be adjusted to a smaller value.

[0039] In the training phase, the descriptors unit 212 associates, for each 3D point \( p_i \), a corresponding set of local 2D feature descriptors \( \{d_{ij}\} \) indicative of a visual appearance of a visual appearance of a 3D object surrounding the 3D point \( p_i \). Specifically, during 3D reconstruction, each 3D point \( p_i \) is natively associated with a set of local 2D feature descriptors \( \{d_{ij}\} \) comprising a set of 2D keypoints \( \{k_{ij}\} \) from a training image \( l_i \) in a track \( j \) representing a projection of a 3D point \( p_j \).

[0040] Localization errors may arise when localizing an image from an uncovered angle as a set of training images \( \{l_i\} \) may not cover a visual appearance of a 3D object surrounding a 3D point \( p_i \) from all possible viewpoint angles. To reduce or eliminate such errors, 2D features are augmented by warping each training image 312 in a set of training images \( \{l_i\} \) to create a set of synthetic views.

[0041] In one embodiment, a training image 312 is warped by placing the image 312 in a virtual hemisphere parameterized in accordance with equation (3) provided below:

\[
(x, y, z) = (\cos(u)\sin(v), \sin(u)\sin(v), \cos(v))
\]  

(3)

wherein \( u \) and \( v \) are points on a surface of the virtual hemisphere surface. Let \( N^T \) denote sampled/target normal directions.
The points u and v are uniformly sampled to obtain sampled/target normal directions $N^T$ in accordance with equation (4) provided below:

\[
N^T = (x_u, y_u, z_u)
\]  

[0042] The training image 312 is rotated to align a normal direction of the image to each sampled/target normal direction of $N^T$, and one or more warped images providing synthetic views are rendered. For each 3D point $p_i$, a corresponding set of local 2D feature descriptors $d_{ij}$ is recomputed based on the warped images and added to the 3D point $p_i$.

[0043] FIG. 5 illustrates an example image alignment, in one or more embodiments. In the deployment phase, the image alignment unit 220 estimates a camera pose of an input image 313 based on at least one learned feature database (e.g., a learned feature database 320 trained by the training system 190 in the training phase) to localize its camera pose, and aligns the input image 313 to a desired camera pose to produce an aligned image 360. For example, if the input image 313 comprises a 2D image of a POS terminal 50, the resulting aligned image 360 comprises a portion of the 2D image of the POS terminal 50 aligned to a desired camera pose.

[0044] Let $(x_k, y_k)$ denote a pair of 2D-3D points, wherein $k = 1, \ldots, N$, and $x_k, y_k$ are 2D/3D points. Let $A$ denote a matrix comprising all unknown camera parameters of camera. Let $X$ and $Y$ denote matrices comprising $x_k$ and $y_k$ in corresponding columns, respectively. The matrices $A$, $X$, and $Y$ satisfy equation (5) provided below:

\[
X = AY
\]

[0045] In one embodiment, matrix $A$ is determined in accordance with equation (6) provided below:

\[
A = XY^T(YY^T)^{-1}
\]

[0046] In one embodiment, the image alignment unit 220 applies a direct linear transformation (DLT) algorithm (e.g., a 6-point DLT algorithm such as P6P) to estimate a camera pose of an input image 313. Alternatively, the image alignment unit 220 applies a Perspective-Three-Point (P3P) algorithm if the input image 313 has an exchangeable image file format (Exif) tag that includes focal length.

[0047] 3D objects from different fine-grained object categories share similar shapes and scales. As such, the image alignment unit 220 determines a single 3D rectangular surface 355 to utilize as an approximated bounding box for all fine-grained object categories. To align an input image 313, the image alignment unit 220 projects the 3D rectangular surface 355 onto the input image 313 based on camera information (e.g., matrices $K$, $R$, and $T$), and obtains four projected 2D corners $O_1$, $O_2$, $O_3$, and $O_4$. The image alignment unit 220 transforms the four projected 2D corners $O_1$, $O_2$, $O_3$, and $O_4$ to a 2D rectangle 365. A resulting aligned image 360 comprises a cropped image inside the 2D rectangle 365, wherein the cropped image is a portion of the input image 313 that includes the 3D object.

[0048] In one embodiment, in the deployment phase, the image alignment unit 220 computes a group of camera pose candidates for each probe image 313 utilizing feature databases 320 for all fine-grained object categories. To generate the largest percentage of inlier points, the best camera pose candidate is selected from the group. In one embodiment, the best camera pose candidate is selected based on a measurement of similarity.

[0049] Let $F$ denote a learned 2D convolutional filter bank (e.g., a learned filter bank 197 trained by the training system 190 in the training phase). The learned filter bank $F$ comprises a set of learned convolutional kernels $f_k$, wherein $F = \{f_k\}$, and $1 \leq k \leq K$. Without loss of generality, the number of learned convolutional kernels $f_k$ included in the learned filter bank $F$ is set to $K$. The learned filter bank $F$ is compact and has a moderate model complexity (i.e., total number of parameters $\Theta(10^k)$).

[0050] In one embodiment, the number of learned convolutional kernels $f_k$ included in the learned filter bank $F$ is set to $K = 256$. Each learned convolutional kernel $f_k$ represents a semantically meaningful learned pattern having size $11 \times 11$.

[0051] The set of learned convolutional kernels $f_k$ represent significant variations, such as different orientations, frequencies, and distinct structures. The learned filter bank $F$ may be used to characterize higher-order image statistics by decomposing each training image $x_i$ of the set $X$ as a sum of a series of sparse feature maps $e_i \in E^i$ convolved with learned convolutional kernels $f_k$ from the learned filter bank $F$.

[0052] As stated above, in one embodiment, in the training phase, the feature learning unit 234 of the training system 190 applies CSC to learn a filter bank $F$. Let $X$ denote a set of $N$ training images 312, wherein $X = (x_i)$, and $1 \leq i \leq N$. 
Each training image $x_i$ of the set $X$ comprises a 2D image with dimension $p \times q$. In one embodiment, the feature learning unit 234 applies CSC to solve equation (7) provided below:

$$
\min_{F, E} \mathcal{L} = \sum_{i=1}^{N} \left\{ \left\| x_i - \sum_{k=1}^{K} f_k \ast e_k^i \right\|_F^2 + \alpha \sum_{k=1}^{K} \left\| e_k^i \right\|_1 \right\}
$$

s.t. $\left\| f_k \right\|_2^2 = 1, \forall k = 1, \ldots, K$

(7),

wherein a first term of equation (7) represents a reconstruction error, a second term of equation (7) represents a $l_1$-norm penalty, $\alpha$ represents a regularization constant, $*$ represents a discrete convolution operator, and $f_k$ and $e_k^i$ are restricted to representing a unit energy to avoid trivial solutions. The learned filter bank $F$ is constructed by balancing the reconstruction error and the $l_1$-norm penalty.

[0053] In one embodiment, the feature learning unit 234 applies a general iterative method to optimize equation (7). For example, the feature learning unit 234 may apply an iterative Shrinkage Thresholding algorithm (ISTA) to compute sparse feature maps $e_k^i$. A stochastic gradient descent may be used to update the learned filter bank $F$ based on training one sample at a time. As another example, the feature learning unit 234 applies a Fast Iterative Shrinkage Thresholding algorithm (FISTA) to solve a $l_1$-norm regularized least squares problems. FISTA has quadratic convergence properties.

[0054] FIG. 6 illustrates an example learned filter bank 400, in one or more embodiments. The filter bank 400 may be an example learned filter bank 197 trained by the training system 190 in the training phase. The filter bank 400 comprises a set of learned convolutional kernels 401 that may include, but are not limited to, edge primitive filters 401A at various angles and filters representing complex feature patterns, such as circular shaped filters 401B, corner structure filters 401C, other types of filters used to extract these salient features for object recognition.

[0055] FIG. 7 illustrates an example single-layer feature extraction scheme, in one or more embodiments. As stated above, in the deployment phase, the feature extraction unit 231 of the object recognition system 200 utilizes a single-layer feature extraction scheme that provides both low-level feature representation and high-level feature representation of an object. For example, instead of utilizing a multi-layer deep neural network, the single-layer feature extraction scheme may be implemented as a pipeline 235. In one embodiment, before applying feature extraction, all input data (e.g., each aligned image 360) is resized into the same size (e.g., $100 \times 100$). A convolution layer 231A of the pipeline 235 is configured to convolve each aligned image 360 with the learned convolutional kernels $f_k$ from a learned filter bank $F$ (e.g., a learned filter bank 197 trained by the training system 190 in the training phase). During convolution, each aligned image 360 is decomposed as a sum of a series of sparse feature maps $e_k^i \in E^i$ convolved with learned convolutional kernels $f_k$ from the learned filter bank $F$.

[0056] After convolution, the sparse feature maps $e_k^i$ are processed by subsequent layers of the pipeline 235. In one embodiment, the pipeline 235 comprises the following three cascaded layers: (1) an element-wise absolute value rectification (ABS) layer 231B, (2) a local contrast normalization (LCN) layer 231C, and (3) a max-pooling (MP) layer 231D. The ABS layer 231B is configured to compute an absolute value element-wise for a given feature map $m_k$, where $y_k^{AB} = |e_k|^i$ represents the k-th feature map. The LCN layer 231C is configured to apply local subtractive and divisive operations within each feature map to enhance stronger feature responses and suppress weaker ones across the feature map.

[0057] In one embodiment, the pipeline 235 may not include the ABS layer 231B and the LCN layer 231C (i.e., ABS and LCN are non-essential processing steps). Not having to perform ABS and LCN increases efficiency of the pipeline 235, thereby reducing overall computation time, especially for input data comprising a high-resolution image. As object recognition on images with abnormal lighting conditions becomes much more challenging without ABS and LCN, however, pre-processing steps may be incorporated at the beginning of the pipeline 235, such as an illumination and resolution detection layer. Multiple algorithms may be applied during these pre-processing steps, such as, but not limited to, measuring brightness of an input image as a reference of a condition of illumination.

[0058] In one embodiment, after LCN, the MP layer 231D applies max-pooling to down-sample feature maps and create position invariance over larger local regions. Max-pooling may be applied to select invariant features and improve generalization performance. One or more resulting feature maps 240 are obtained after max-pooling.

[0059] Let $Y^m$ denote one or more features from a resulting feature map 240 after max-pooling, wherein $Y^m = \{y_k^m\}$, and $1 \leq k \leq K$. Each feature $Y^m$ may be referred to as a Hierarchical Learning Feature (HLF).

[0060] Let $Y^m$ denote one or more features from a resulting feature map 240 after max-pooling, wherein $Y^m = \{y_k^m\}$, and $1 \leq k \leq K$. Each feature $Y^m$ may be referred to as a Hierarchical Learning Feature (HLF).

[0061] As stated above, in one embodiment, in the training phase, the feature learning unit 234 of the training system 190 applies a manifold learning algorithm to learn a projection matrix for reducing feature dimensions. Let $W$ denote a low-dimensional space, and $P$ denote a projection matrix for transforming original features $Y^m$ to the low-dimensional space $W$. In one embodiment, the feature learning unit 234 determines a projection matrix $P$ for transforming original features $Y^m$ to a low-dimensional space $W$ satisfying equation (8) provided below:
wherein \( P \in R^{d\times 1} \), \( \gamma^m \in R^{d\times N} \), \( P^T \gamma^m \in R^{1\times N} \), \( 1 \geq d \), and \( R \) denotes a high dimensional data space.

In the deployment phase, the manifold learning unit 232 applies a learned projection matrix (e.g., a learned projection matrix 195 trained by the training system 190 in the training phase) to reduce feature dimensions for original features \( \gamma^m \). Specifically, the manifold learning unit 232 projects a first low-dimensional space \( 245 \) from a second dimensional space, wherein the second dimensional space has more dimensions than the first dimensional space. This enables fine-grained object classification on devices with limited resources (e.g., mobile devices).

In the deployment phase, a resulting low-dimensional space \( 245 \) is directly fed into the classifier 233. In one embodiment, the classifier 233 utilizes a learned classification model (e.g., a learned classification model 196 trained by the training system 190 in the training phase) to map a feature of the low-dimensional space \( 245 \) to a corresponding fine-grained object category.

Let \( \theta \) denote an input feature vector, let \( y \) denote a label for the feature vector \( \theta \), and let \( k(\theta, y) \) denote a histogram intersection kernel 250. In one embodiment, the classifier 233 selects a histogram intersection kernel \( k(\theta, y) \) satisfying equation (9) provided below:

\[
k(\theta, y) = \sum_{i=1}^{N} \min(\theta(i), y(i)) \min(\theta(i), y(i))
\]

The classifier 233 is configured to perform classification based on an evaluation of \( \text{sign}(d(\theta)) \), wherein \( d(\theta) \) is represented in accordance with equation (10) provide below:

\[
d(\theta) = \sum_{j=1}^{N} a_j y_j k(\theta, \theta_j) + b
\]

wherein \( y_j \) denotes a label for feature \( \theta_j \) of the feature vector \( \theta \), and \( a \) and \( b \) are constants.

FIG. 8 is a flowchart of an example process 800 for fine-grained object recognition, in one or more embodiments. In process block 801, estimate a camera pose of an input image, where the input image comprises an image of a fine-grained object. In process block 802, align the input image to a desired camera pose based on a feature database. In process block 803, classify the object based on the aligned input image and a learned classification model.

In one embodiment, process blocks 801-803 may be performed by at least the image alignment unit 220 and the fine-grained object classification unit 230 of the object recognition system 200.

FIG. 9 is a high-level block diagram showing an information processing system comprising a computer system 600 useful for implementing the disclosed embodiments. Computer system 600 may be incorporated in a display device 300 or a server device 210. The computer system 600 includes one or more processors 601, and can further include an electronic display device 602 (for displaying video, graphics, text, and other data), a main memory 603 (e.g., random access memory (RAM)), storage device 604 (e.g., hard disk drive), removable storage device 605 (e.g., removable storage drive, removable memory module, a magnetic tape drive, optical disk drive, computer readable medium having stored therein computer software and/or data), viewer interface device 606 (e.g., keyboard, touch screen, keypad, pointing device), and a communication interface 607 (e.g., modem, a network interface (such as an Ethernet card), a communications port, or a PCMCIA slot and card). The communication interface 607 allows software and data to be transferred between the computer system and external devices. The system 600 further includes a communications infrastructure 608 (e.g., a communications bus, cross-over bar, or network) to which the aforementioned devices/modules 601 through 607 are connected.

Information transferred via communications interface 607 may be in the form of signals such as electronic, electromagnetic, optical, or other signals capable of being received by communications interface 607, via a communication link that carries signals and may be implemented using wire or cable, fiber optics, a phone line, a cellular phone link, an radio frequency (RF) link, and/or other communication channels. Computer program instructions representing the block diagram and/or flowcharts herein may be loaded onto a computer, programmable data processing apparatus, or processing devices to cause a series of operations performed thereon to generate a computer implemented process. In one embodiment, processing instructions for process 800 (FIG. 8) may be stored as program instructions on the memory 603, storage device 604 and the removable storage device 605 for execution by the processor 601.

Embodiments have been described with reference to flowchart illustrations and/or block diagrams of methods, apparatus (systems) and computer program products. Each block of such illustrations/diagrams, or combinations thereof,
can be implemented by computer program instructions. The computer program instructions when provided to a processor produce a machine, such that the instructions, which execute via the processor create means for implementing the functions/operations specified in the flowchart and/or block diagram. Each block in the flowchart /block diagrams may represent a hardware and/or software module or logic. In alternative implementations, the functions noted in the blocks may occur out of the order noted in the figures, concurrently, etc.

[0071] The terms "computer program medium," "computer usable medium," "computer readable medium," and "computer program product," are used to generally refer to media such as main memory, secondary memory, removable storage drive, a hard disk installed in hard disk drive, and signals. These computer program products are means for providing software to the computer system. The computer readable medium allows the computer system to read data, instructions, messages or message packets, and other computer readable information from the computer readable medium. The computer readable medium, for example, may include non-volatile memory, such as a floppy disk, ROM, flash memory, disk drive memory, a CD-ROM, and other permanent storage. It is useful, for example, for transporting information, such as data and computer instructions, between computer systems. Computer program instructions may be stored in a computer readable medium that can direct a computer, other programmable data processing apparatus, or other devices to function in a particular manner, such that the instructions stored in the computer readable medium produce an article of manufacture including instructions which implement the function/act specified in the flowchart and/or block diagram block or blocks.

[0072] As will be appreciated by one skilled in the art, aspects of the embodiments may be embodied as a system, method or computer program product. Accordingly, aspects of the embodiments may take the form of an entirely hardware embodiment, an entirely software embodiment (including firmware, resident software, micro-code, etc.) or an embodiment combining software and hardware aspects that may all generally be referred to herein as a "circuit," "module" or "system." Furthermore, aspects of the embodiments may take the form of a computer program product embodied in one or more computer readable medium(s) having computer readable program code embodied thereon.

[0073] Any combination of one or more computer readable medium(s) may be utilized. The computer readable medium may be a computer readable storage medium. A computer readable storage medium may be, for example, but not limited to, an electronic, magnetic, optical, electromagnetic, infrared, or semiconductor system, apparatus, or device, or any suitable combination of the foregoing. More specific examples (a non-exhaustive list) of the computer readable storage medium would include the following: an electrical connection having one or more wires, a portable computer diskette, a hard disk, a random access memory (RAM), a read-only memory (ROM), an erasable programmable read-only memory (EPROM or Flash memory), an optical fiber, a portable compact disc read-only memory (CD-ROM), an optical storage device, a magnetic storage device, or any suitable combination of the foregoing. In the context of this document, a computer readable storage medium may be any tangible medium that can contain, or store a program for use by or in connection with an instruction execution system, apparatus, or device.

[0074] Computer program code for carrying out operations for aspects of one or more embodiments may be written in any combination of one or more programming languages, including an object oriented programming language such as Java, Smalltalk, C++ or the like and conventional procedural programming languages, such as the "C" programming language or similar programming languages. The program code may execute entirely on the user's computer, partly on the user's computer, as a stand-alone software package, partly on the user's computer and partly on a remote computer or entirely on the remote computer or server. In the latter scenario, the remote computer may be connected to the user's computer through any type of network, including a local area network (LAN) or a wide area network (WAN), or the connection may be made to an external computer (for example, through the Internet using an Internet Service Provider).

[0075] Aspects of one or more embodiments are described above with reference to flowchart illustrations and/or block diagrams of methods, apparatus (systems) and computer program products. It will be understood that each block of the flowchart illustrations and/or block diagrams, and combinations of blocks in the flowchart illustrations and/or block diagrams, can be implemented by computer program instructions. These computer program instructions may be provided to a special purpose computer, or other programmable data processing apparatus to produce a machine, such that the instructions, which execute via the processor of the computer or other programmable data processing apparatus, create means for implementing the functions/acts specified in the flowchart and/or block diagram block or blocks.

[0076] These computer program instructions may also be stored in a computer readable medium that can direct a computer, other programmable data processing apparatus, or other devices to function in a particular manner, such that the instructions stored in the computer readable medium produce an article of manufacture including instructions which implement the function/act specified in the flowchart and/or block diagram block or blocks.

[0077] The computer program instructions may also be loaded onto a computer, other programmable data processing apparatus, or other devices to cause a series of operational steps to be performed on the computer, other programmable apparatus or other devices to produce a computer implemented process such that the instructions which execute on the computer or other programmable apparatus provide processes for implementing the functions/acts specified in the flowchart and/or block diagram block or blocks.

[0078] The flowchart and block diagrams in the Figures illustrate the architecture, functionality, and operation of possible
implementations of systems, methods, and computer program products according to various embodiments. In this regard, each block in the flowchart or block diagrams may represent a module, segment, or portion of instructions, which comprises one or more executable instructions for implementing the specified logical function(s). In some alternative implementations, the functions noted in the block may occur out of the order noted in the figures. For example, two blocks shown in succession may, in fact, be executed substantially concurrently, or the blocks may sometimes be executed in the reverse order, depending upon the functionality involved. It will also be noted that each block of the block diagrams and/or flowchart illustration, and combinations of blocks in the block diagrams and/or flowchart illustration, can be implemented by special purpose hardware-based systems that perform the specified functions or acts or carry out combinations of special purpose hardware and computer instructions.

References in the claims to an element in the singular is not intended to mean "one and only" unless explicitly so stated, but rather "one or more." All structural and functional equivalents to the elements of the above-described exemplary embodiment that are currently known or later come to be known to those of ordinary skill in the art are intended to be encompassed by the present claims.

The terminology used herein is for the purpose of describing particular embodiments only and is not intended to be limiting of the invention. As used herein, the singular forms "a," "an" and "the" are intended to include the plural forms as well, unless the context clearly indicates otherwise. It will be further understood that the terms "comprises" and/or "comprising," when used in this specification, specify the presence of stated features, integers, steps, operations, elements, and/or components, but do not preclude the presence or addition of one or more other features, integers, steps, operations, elements, components, and/or groups thereof.

The corresponding structures, materials, acts, and equivalents of all means or step plus function elements in the claims below are intended to include any structure, material, or act for performing the function in combination with other claimed elements as specifically claimed. The invention is defined by the appended claims.

Claims

1. A computer implemented method comprising:
   estimating (801) a camera pose of an input image, wherein the input image comprises an image of an object; aligning (802) the input image to a desired camera pose.
   wherein the alignment of the input image is based on a feature database (320) comprising a set of sparse 3D points representing sparse geometry of the shape of the object, wherein each 3D point of the feature database is associated with a corresponding set of local multi-dimensional feature descriptors indicative of a visual appearance of the object about the 3D point; and classifying (803) the object based on the resulting aligned input image, characterized in that the classifying comprises:
   applying a single-layer feature extraction scheme that provides both low-level feature representation and high-level feature representation of the object, wherein a convolution layer is configured to convolve each aligned image with learned convolutional kernels \( f_k \) from a learned filter bank \( F \), wherein the filter bank \( F \) is learned in a training phase by applying Convolutional Sparse Coding, CSC, to solve the following equation:

\[
\text{min}_{F, E} \mathcal{L} = \sum_{i=1}^{N} \left\{ \left\| x_i - \sum_{k=1}^{K} f_k \ast e_{ik} \right\|_F^2 + \alpha \sum_{k=1}^{K} \left\| e_{ik} \right\|_1 \right\}
\]

s.t. \( \left\| f_k \right\|_2^2 = 1, \forall k = 1, \ldots, K \)

wherein \( X \) denotes a set of \( N \) training images, wherein \( X = \{x_i\} \), and \( 1 \leq i \leq N \), wherein each training image \( x_i \) of the set \( X \) comprises a 2D image with dimension \( p \times q \), wherein \( \alpha \) represents a regularization constant, \( \ast \) represents a discrete convolution operator, \( e_{ik} \), represents the extracted sparse feature maps, and \( f_k \) and \( e_{ik} \) are restricted to representing a unit energy; and
   applying a learned projection matrix for transforming the extracted features of the resulting aligned input image to a low-dimensional space.

2. The method of claim 1, wherein the set of sparse 3D points is based on a set of images including the object, and the set of images are captured from different camera poses to illustrate different illumination changes and back-
3. The method of claim 2, wherein a portion of the object in each image of the set of images triangulates to a same 3D point of the feature database.

4. A system (600), comprising:
   at least one processor (601); and
   a non-transitory processor-readable memory device (603) storing instructions that when executed by the at least one processor causes the at least one processor to perform the computer-implemented method of any one of claims 1 to 3.

5. A non-transitory computer readable storage medium including instructions to perform the computer-implemented method of any one of claims 1 to 3.

Patentansprüche

1. Computerimplementiertes Verfahren, umfassend:
   Abschätzen (801) einer Kameraposition eines Eingabebilds, wobei das Eingabebild ein Bild eines Objekts umfasst;
   Ausrichten (802) des Eingabebilds auf eine gewünschte Kameraposition, wobei die Ausrichtung des Eingabebilds auf einer Merkmalsdatenbank (320) basiert, die einen Satz spärlicher 3D-Punkte umfasst, die die spärlche Geometrie der Objektform darstellen, wobei jeder 3D-Punkt der Merkmalsdatenbank einem entsprechenden Satz lokaler mehrdimensionaler Merkmalsdeskriptoren zugeordnet ist, die ein visuelles Erscheinungsbild des Objekts um den 3D-Punkt herum anzeigen; und
   Klassifizieren (803) des Objekts basierend auf dem resultierenden ausgerichteten Eingabebild, dadurch gekennzeichnet, dass das Klassifizieren Folgendes umfasst:
   Anbringen eines einschichtigen Merkmalsextraktionsschemas, das sowohl eine Merkmalsdarstellung auf niedriger Ebene als auch eine Merkmalsdarstellung auf hoher Ebene des Objekts bereitstellt, wobei eine Faltungsschicht dazu konfiguriert ist, um jedes ausgerichtete Bild mit gelernten Faltungskernen \( f_k \) aus einer gelernten Filterbank \( F \) zu falten, wobei die Filterbank \( F \) in einer Trainingsphase durch Anwenden von Convolutional Sparse Coding (CSC), gelernt wird, um die folgende Gleichung zu lösen:

\[
\min_{F, E} \mathcal{L} = \sum_{k=1}^{K} \left\| x_i - \sum_{k=1}^{K} f_k * e_k \right\|_F^2 + \alpha \sum_{k=1}^{K} \| e_k \|_1
\]

wobei \( X \) einen Satz von \( N \)-Trainingsbildern bezeichnet, wobei \( X = \{ x_i \}, \) und \( 1 \leq i \leq N, \) wobei jedes Trainingsbild \( x_i \) des Satzes \( X \) ein 2D-Bild mit der Dimension \( p \times q \) umfasst, wobei \( \alpha \) eine Regularisierungskonstante darstellt, \( * \) einen diskreten Faltungsoperator repräsentiert, \( e_k \) die extrahierten spärlichen Feature-Maps repräsentiert, und \( f_k \) und \( v_k \) darauf beschränkt sind, eine Einheitsenergie darzustellen; und
   Anbringen einer gelerneten Projektionsmatrix zum Transformieren der extrahierten Merkmale des resultierenden ausgerichteten Eingabebildes in einen niedrigdimensionalen Raum.

2. Verfahren nach Anspruch 1, wobei der Satz spärlicher 3D-Punkte auf einem Satz von Bildern basiert, die das Objekt enthalten, und der Satz von Bildern aus verschiedenen Kamerapositionen aufgenommen wird, um verschiedene Beleuchtungsänderungen und Hintergründe des Objekts zu veranschaulichen.

4. System (600), umfassend:

mindestens einen Prozessor (601); und
eine nichtflüchtige prozessorlesbare Speichervorrichtung (603), die Anweisungen speichert, die, wenn sie von
dem mindestens einen Prozessor ausgeführt werden, den mindestens einen Prozessor veranlassen, das com-
puterimplementierte Verfahren nach einem der Ansprüche 1 bis 3 durchzuführen.

5. Nichtflüchtiges computerlesbares Speichermedium, das Anweisungen zum Durchführen des computerimplemen-
tierten Verfahrens nach einem der Ansprüche 1 bis 3 enthält.

Revendications

1. Procédé mis en œuvre par ordinateur comprenant :
estimer (801) une pose de caméra d’une image d’entrée, où l’image d’entrée comprend une image d’un objet ;
aligner (802) l’image d’entrée à une pose de caméra souhaitée.
où l’alignement de l’image d’entrée est basé sur une base de données de caractéristiques (320) comprenant
un ensemble de points 3D épars représentant une géométrie épars de la forme de l’objet, où chaque point
3D de la base de données de caractéristiques est associé à un ensemble correspondant de descripteurs de
caractéristiques multidimensionnelles locales indiquant un aspect visuel de l’objet autour du point 3D ; et
classer (803) l’objet sur la base de l’image d’entrée alignée résultante,
caractérisé en ce que

appliquer un schéma d’extraction de caractéristiques monocouche qui fournit à la fois une représentation
de caractéristiques de bas niveau et une représentation de caractéristiques de haut niveau de l’objet, où
une couche de convolution est configurée pour convoluer chaque image alignée avec des noyaux de
convolution appris fk à partir d’une banque de filtres F apprise, où la banque de filtres F est apprise dans
une phase d’apprentissage en appliquant un codage épars convolutif, CSC, pour résoudre l’équation
suivante :

\[
\min_{F, E} \mathcal{L} = \sum_{i=1}^{N} \left\{ \left\| x_i - \sum_{k=1}^{K} f_k \ast e_k^i \right\|^2_F + \alpha \sum_{k=1}^{K} \left\| e_k^i \right\|_1 \right\}
\]

s.t. \( \left\| f_k \right\|^2_2 = 1, \forall k = 1, \ldots, K \)

où X indique un ensemble de N images d’apprentissage, où X = (x_i), et 1 ≤ i ≤ N, où chaque image
d’apprentissage x_i de l’ensemble X comprend une image 2D ayant des dimensions p x q, où α représente
une constante de régularisation, * représente un opérateur de convolution discret, e_k^i, représente les cartes
de caractéristiques éparses extraites,
et f_k et e_k sont limités à la représentation d’une énergie unitaire ; et
appliquer une matrice de projection apprise pour transformer les caractéristiques extraites de l’image d’en-
trée alignée résultante en un espace de faibles dimensions.

2. Procédé selon la revendication 1, où l’ensemble de points 3D épars est basé sur un ensemble d’images incluant
l’objet, et l’ensemble d’images est capturé à partir de différentes poses de caméra pour illustrer différents change-
ments d’éclairage et arrière-plans de l’objet.

3. Procédé selon la revendication 2, où une partie de l’objet dans chaque image de l’ensemble d’images est triangulée
à un même point 3D de la base de données de caractéristiques.

4. Système (600), comprenant :

au moins un processeur (601) ; et
un dispositif de mémoire lisible par processeur non transitoire (603) stockant des instructions qui, lorsqu’elles sont exécutées par l’au moins un processeur, amènent l’au moins un processeur à effectuer le procédé mis en œuvre par ordinateur selon l’une quelconque des revendications 1 à 3.

5. Support de stockage lisible par ordinateur non transitoire incluant des instructions pour effectuer le procédé mis en œuvre par ordinateur selon l’une quelconque des revendications 1 à 3.
[Fig. 2]

190

TRAINING SYSTEM

DATABASE BUILDER

SfM 211

DESCRIPTORS 212

210

FEATURE LEARNING 234

PROJECTION MATRIX 195

CLASSIFICATION MODEL 196

FILTER BANK 197

IMAGE DATABASE

2D IMAGES 312

310

CATEGORY_1

IMAGE DATABASE

2D IMAGES 312

310

CATEGORY_N
OBJECT RECOGNITION SYSTEM

IMAGE ALIGNMENT
- LOCALIZATION 221
- ALIGNMENT 222
- 220

FINE-GRAINED OBJECT CLASSIFICATION
- FEATURE EXTRACTION 231
- MANIFOLD LEARNING 232
- CLASSIFIER 233
- 230

FEATURE DATABASE
- 3D POINT 322
- CATEGORY \(_1\)
- DESCRIPTORS
- 2D FEATURE 332
- 3D POINT \(_1\)

FEATURE DATABASE
- 3D POINT 322
- CATEGORY \(_n\)
- DESCRIPTORS
- 2D FEATURE 332
- 3D POINT \(_m\)
[Fig. 8]

800

- Estimate a Camera Pose of an Input Image, Where the Input Image Comprises an Image of a Fine-grained Object

801

- Align the Input Image to a Desired Camera Pose Based on a Feature Database

802

- Classify the Object Based on the Aligned Input Image and a Learned Classification Model

803
REFERENCES CITED IN THE DESCRIPTION

This list of references cited by the applicant is for the reader’s convenience only. It does not form part of the European patent document. Even though great care has been taken in compiling the references, errors or omissions cannot be excluded and the EPO disclaims all liability in this regard.

Patent documents cited in the description