Pairwise Decomposition of Image Sequences for Active Multi-View Recognition

Edward Johns, Stefan Leutenegger and Andrew J. Davison
Dyson Robotics Laboratory at Imperial College, Department of Computing, Imperial College London, UK

Abstract

A multi-view image sequence provides a much richer capacity for object recognition than from a single image. However, most existing solutions to multi-view recognition typically adopt hand-crafted, model-based geometric methods, which do not readily embrace recent trends in deep learning. We propose to bring Convolutional Neural Networks to generic multi-view recognition, by decomposing an image sequence into a set of image pairs, classifying each pair independently, and then learning an object classifier by weighting the contribution of each pair. This allows for recognition over arbitrary camera trajectories, without requiring explicit training over the potentially infinite number of camera paths and lengths. Building these pairwise relationships then naturally extends to the next-best-view problem in an active recognition framework. To achieve this, we train a second Convolutional Neural Network to map directly from an observed image to next viewpoint. Finally, we incorporate this into a trajectory optimisation task, whereby the best recognition confidence is sought for a given trajectory length. We present state-of-the-art results in both guided and unguided multi-view recognition on the ModelNet dataset, and show how our method can be used with depth images, greyscale images, or both.

1. Introduction

Consider the scenario in Figure 1. What trajectory should the camera move around the object in order to achieve the highest recognition confidence in a given time? For practical tasks, recognition from a *multi-view* image sequence is a more realistic setting than the single-image recognition tasks typically addressed in computer vision, and controlling a camera *actively* for efficient recognition has great significance in real-world applications, where time or power constraints become realities. For example, a robot rotating an object before its eyes, or a mobile robot semantically mapping a room, benefit from efficient solutions.

Traditionally, multi-view object recognition has been achieved by building up compositions of hand-crafted features shared across viewpoints, and finding correspondences between a test image and the learned models [23, 36, 28]. However, recent trends in Convolutional Neural Networks (CNNs) [21, 34] have seen attention in single-view object recognition move away from these explicit, hand-modelled, geometric solutions, and towards end-to-end learning ideologies which inject fewer assumptions into the learned object models. Recently, the introduction of the ModelNet dataset of 3D CAD meshes [39] provided data of sufficient magnitude for training deep networks with images covering the full sphere of viewpoints over an object, enabling view synthesis without the need for laborious manual labelling of each image [16]. It was subsequently shown that rendering these meshes as synthetic greyscale images, and classifying objects in a view-based manner with a CNN architecture acting over a fixed trajectory, achieved state-of-the-art results for multi-view recognition [35]. However, extending this to generalised recognition over trajectories of

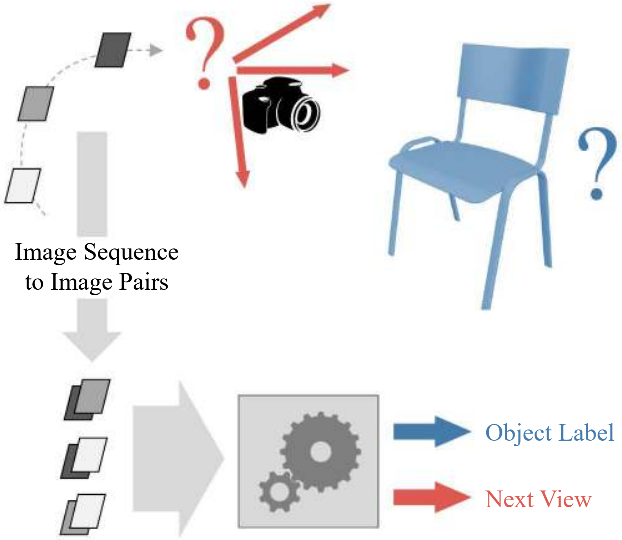

Figure 1: We propose a method for multi-view object recognition, by decomposing an image sequence into a set of image pairs. Training on these pairs then allows for recognition and trajectory planning, without the need to train directly over the infinite possible number of camera paths that may exist.

arbitrary paths and lengths is not readily adopted by traditional CNN architectures, due to the need for fixed-length input data.

1.1. CNNs for Generalised Multi-View Recognition

One solution to multi-view recognition with CNNs would be to simply concatenate all observed images into a single input to a network. However, this would require intractable training due to the large size of each input, but more importantly, due to the need to train over every possible path of all possible lengths, which is of potentially infinite scale. We propose to address this by relaxing the joint model over images and decomposing an image sequence into a set of pairs, one for every pair of images across the sequence. Pairwise representations of full distributions have been popular in computer vision for learning distributions of local features [19] and parts [10], and we migrate this idea from the image space domain to the temporal domain. Given this decomposition, a CNN is then trained on a fixed-length input consisting of the image pair, together with the relative pose between the associated viewpoints. To achieve classification of the full sequence, an ensemble framework is adopted, with weighting to increase the contribution of those image pairs which cover a more informative set of poses.

The problem then shifts to active recognition, with the aim of determining along which trajectory the camera should move, in order to achieve the best recognition accuracy in a given number of images. This is often presented as a Next-Best-View (NBV) prediction, where the mutual information is determined between the class probability distribution and each potential next view. However, this typically requires learning a generative model of the object and synthesising new views as an intermediate step. We propose to learn NBV prediction with a more powerful discriminative model, training a second CNN to map directly from an observed image to the rotation angle over which the camera should subsequently move.

Finally, we extend our NBV prediction to a full trajectory-optimisation framework, where we consider all possible images that can acquired along a trajectory as contributions, rather than simply following a sequence of NBV images as is often employed. To achieve this, we train a third CNN in a similar manner to the above NBV CNN, but training for regression to a recognition confidence score for all possible next viewpoints, rather then classification for the overall best viewpoint. As the image sequence evolves, all unvisited viewpoints accumulate scores based on the newly-observed images, and the optimum trajectory is chosen as the one which maximises the summation of these scores.

1.2. Contributions

In this paper, we present three key technical contributions all based on powerful CNN learning:

1. Multi-view object recognition over arbitrary camera trajectories by training only on image pairs,
2. Discriminatively-trained Next-Best-View prediction directly from an input image to the next viewpoint,
3. Trajectory optimisation by considering the impact of all observable images along the sequence.

All three contributions achieve state-of-the-art results in their respective benchmarks on the ModelNet dataset [39].

2. Related Work

View-Based Multi-View Recognition In its simplest form, the view-based approach aims to add viewpoint tolerance to a 2D image of an object, such as with viewpoint-invariant local descriptors [27, 29] or deformation-tolerant global descriptors [6]. Given training images across multiple viewpoints, a more stable set of features can be found by tracking those which are shared across multiple views and clustering images accordingly [23], or by learning their relative 2D displacements as the viewpoint changes, both with hard constraints for rigid bodies [17, 18] and flexible constraints for deformable bodies [11, 10]. To add further fidelity to the true underlying object geometry, these 2D image elements can also be embedded within an implicit 3D model [36, 22, 28]. If multiple views are available at testing, images can be combined and treated as a single, larger image [31], an approach which can also be addressed in two stages, by processing the individual images first to reduce the search space [5].

Recently, CNN architectures have been extended to allow for recognition from image sequences using a single network, by max pooling across all viewpoints [35], or by unwrapping an object shape into a panorama and max pooling across each row [33]. However, both these methods assume that a fixed-length image sequence is provided during both training and testing, and hence are unsuitable for generalised multi-view recognition.

Shape-Based Multi-View Recognition Rather than modelling an object as a set of views with 2D features, an explicit 3D shape can be learned from reconstruction [37] or provided by CAD models [39], and subsequently matched to from depth images [13], 3D reconstructions [1], or partial reconstructions with shape completion [12, 39]. Shape descriptors include distributions of local surface properties [14, 32], spherical harmonic functions over voxel grids [24], and 3D local invariant features [25]. Recently, CNNs have been applied to 3D shapes by representing them as 3D occupancy grids, and building generative [39] or discriminative [26] networks.

As of now however, CNNs with 2D view-based methods [35] have outperformed their counterpart 3D voxel-based methods [39, 26], and we therefore adopt the 2D approach in our work. However, it is not yet clear whether this greater performance arises from the superior abundance of 2D image data for pre-training deep networks, or the naturally more efficient representation of 2D than 3D in standard CNN architectures.

Active Recognition Methods for active recognition typically learn a generative model of the object, predict the object appearance from unvisited viewpoints, and select views based on a measure of entropy reduction. [38] modelled objects as a 3D cloud of SIFT features, moving the camera to the view which would reveal the greatest number of features which have not yet been observed. A similar method was proposed in [2] for guided mapping and robot navigation. The incorporation of active recognition into a Random Forests framework was presented in [8], whereby each decision tree encodes both object classification and viewpoint selection. Recently, the ShapeNets framework of [39] proposed to model objects as a voxel grid, and learn a generative model based on Convolutional Deep Belief Networks to allow for view synthesis from unseen viewpoints.

However, these methods do not take into account the images acquired along a sequence towards the chosen next view. In [20], this was incorporated during active object reconstruction by visiting a sequence of actively-selected views, but reconstructing the object based on the entire image sequence that is observed between the views. For recognition, Partially Observable Markov Decision Processes (POMDPs) [9] have seen success in optimising a trajectory for a particular task, although these require generative modelling rather than direct discriminative learning as we propose in our method. Finally, recurrent CNNs have recently been shown to be effective for active recognition from image sequences [15], and we believe that this approach has exciting future potential.

3. Multi-View Object Recognition

3.1. Dataset

We train and test our proposed methods on the ModelNet dataset of 3D CAD meshes [39], which provides multi-view training data of sufficient scale for training deep networks. As in [39, 35], we discretise viewpoints into distinct steps, but whereas in these works rotations are constrained to being around the gravity vector, we relax this and allow rotations around the object's full viewing sphere to enable recognition from arbitrary camera trajectories. The camera pose is defined in spherical coordinates $\{r, \theta, \phi\}$, where r is fixed and θ and ϕ are divided into 30° steps, and the camera is pointed towards the object's centroid. Camera paths then visit viewpoints along the viewing sphere with θ and ϕ either decreasing or increasing by one step, or remaining the same, between viewpoints. For every viewpoint, we render a greyscale image of the object object as with [35], together with a depth image for dual-modality imaging.

For comparisons with [39, 35], we also assume each object to be aligned in its canonical orientation as defined in the ModelNet dataset, although augmenting the training data by rotating models as necessary would allow for relaxation of this prior assumption. As with these works, we also assume the pose of the camera to be known with respect to the object's viewing sphere, whereas in practice this would be achieved by visual tracking or reconstruction, or by use of robot kinematics or other external sensors. Training and testing models are provided as CAD meshes and free from occlusion or clutter, although in practice a detection and segmentation task would precede our pipeline.

3.2. Pairwise Learning

Our proposed multi-view object recognition method requires computing the probability over class labels given a sequence of M views. To allow for flexibility of camera trajectories over all possible paths and lengths, we decompose a sequence into a set of $N = \frac{M(M-1)}{2}$ *view pairs*, denoted $w_1...w_N$. Here, every new view acquired along a sequence forms a new view pair with all existing views in the sequence, and the task then becomes to compute a recognition score over all classes, $f(\mathbf{y}|w_1...w_N)$.

The data for each view pair w_i is composed of three elements: the image x_i^1 from the first view, the image x_i^2 from the second view, and the relative camera pose ψ_i between the two views, such that $w_i = \{x_i^1, x_i^2, \psi_i\}$. For object recognition from a view sequence, we classify each view pair independently, and then weight the contribution from each, discussed further in Section 3.3. In this way, each view pair w_i is processed with a *weak* classifier, with an associated weight λ_i, and a *strong* classifier then computes the weighted average of these for a final distribution of scores over class labels:

$$f(\mathbf{y}|w_1...w_N) = \sum_{i=1}^{i=N} \lambda_i \, p(\mathbf{y}|w_i) \, . \qquad (1)$$

To compute the class probability distribution $p(\mathbf{y}|w_i)$ for each view pair, we designed a CNN architecture, denoted *CNN-1* (see Figure 2), to predict an object class based on the provided view pair. This architecture was inspired by the Siamese CNN [4], which consists of two CNN's running in parallel, each taking in one image from the pair, and with weights shared across both networks. Whilst this architecture is typically used to enforce similarity or dissimilarity between the outputs of the two networks, we use it to reduce the number of parameters to be learned, rather than concatenating the two images into a single input vector and training a larger network. Not only does this allow for efficient training, but also a fair comparison with the state-of-the-art [35] because we are not adding additional capacity to the network, as this has been shown to dramatically improve classification performance [34].

For the convolutional layers of CNN-1, we follow [35] and adopt the VGG-M network [3] with five convolutional layers and three fully-connected layers. The final convolutional layers from the two images are concatenated, together with a vector using one-hot encoding to represent the relative camera pose between the two views. Depending on the available imaging modality, the framework can be used with greyscale or depth images, or both. When both are used, we also concatenate the outputs of the convolutional layers for both modalities. Finally, three fully-connected layers are added after this concatenation, with classification loss computed using softmax and cross-entropy.

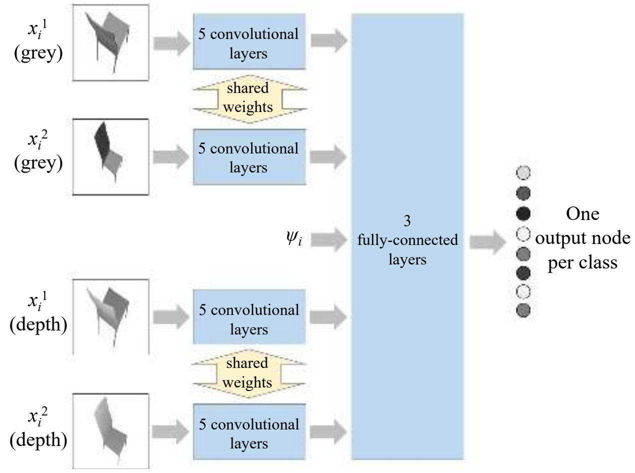

Figure 2: Our CNN-1 architecture for classification of a view pair, for use with greyscale images, depth images, or both.

3.3. Learning the weights

Together with the two images, CNN-1 receives the relative pose of the view pair, and we use this to condition the classifier on how confident its output is likely to be. For example, a pair of images captured from two viewpoints at opposite ends of the viewing sphere may be more likely to be classified correctly than images from two adjacent viewpoints, because the former observes a greater coverage of the object and hence reveals more informative data upon which to make a classification decision.

We use cross entropy to measure the classification confidence for each relative pose, which computes the similarity between the ground truth distribution and the network's output distribution, a richer indicator than simply the classification error. For each relative pose ψ_j in the discretised viewing sphere, the weight λ_j is learned by averaging the cross entropy over all training image pairs whose viewpoints are separated by ψ_j. Then, all viewpoints in the sequence are weighted accordingly in Equation 1 to give greater importance to those view pairs which are likely to be classified correctly. Although the output class distribution of each pair already implicitly exhibits a measure of confidence based on the entropy, this additional weighting acts as a regularisation by injecting prior knowledge of how easily different viewpoint pairings can be separated, independently of the image content.

4. Active Object Recognition

4.1. Next-Best-View Prediction

Given one view of an object, predicting the next view to move the camera to enables an active approach to recognition, by maximising the classification accuracy over a given number of views. Previous works [39] typically address this by building a generative model of the object, predicting the observable image content from all other viewpoints, and choosing the view which, if observed, would reduce the class distribution entropy the most. We propose to solve this in a discriminative end-to-end manner, by training a second CNN, denoted *CNN-2* (see Figure 3), which directly outputs the best viewpoint to move to for any given input image. In this way, NBV prediction is trained in a direct and discriminative manner, with end-to-end learning which bypasses the intermediate step of generative prediction.

As with CNN-1, this network is based on the VGG-M network [3], with 5 convolutional and 3 fully-connected layers. However, rather than outputting a distribution over class labels as with CNN-1, the final layer consists of one node for every relative pose along the viewing sphere. To train CNN-2, every training image x_k is paired with all other images from that same object, and the view pair is processed with CNN-1 to give a class distribution. Then, the view pair is chosen which yields the highest output for the ground truth class. The relative pose associated with this view pair, together with image x_k, then forms a training pair for CNN-2. During testing, a single view is passed through CNN-2, and the output node with the highest value determines the relative pose for the camera's next movement. A series of NBV movements can then be created by iterating this procedure sequentially.

4.2. Trajectory Optimisation

Although this NBV prediction offers a simple solution to active recognition, it does not consider the images that could

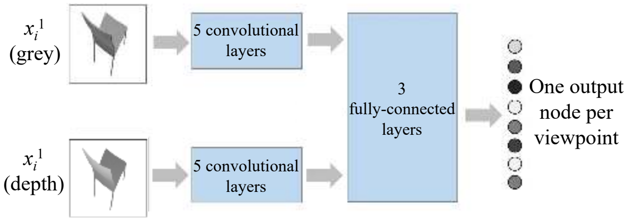

Figure 3: Our CNN-2 architecture for classifying the next-best-view given an input view, based on greyscale images, depth images, or both. This is also our CNN-3 architecture, where the output is a regression to predicted cross entropies over all viewpoints.

be acquired whilst the camera is moving towards the next view, and hence is not a globally-optimum solution. Rather than traversing a sequence of NBVs, recognition efficiency can be maximised by following a path which benefits from the contribution of *all* observable images over the trajectory. Our proposed heuristic is that the optimum trajectory is one whose summation of predicted cross entropies, over all view pairs in the sequence, is smaller than for all other possible trajectories. In this way, we aim to contribute a high classification confidence to Equation 1 for every view pair, rather than only for the view pair formed from the first and last image, as is the case with CNN-2.

This is achieved as follows. For the current trajectory, let us denote the sequence of observed views as the set $v \in \mathcal{V}$. We then maintain a distribution $g(\boldsymbol{u})$ over all unobserved views $u \in \mathcal{U}$ on the viewing sphere, where g_u indicates the cost of visiting view u in the current trajectory. g_u is defined as the sum of predicted cross entropies based on CNN-1, for all view pairs formed between u and the set $\mathcal{V}$. As each new view is observed and added to the sequence, $g(\boldsymbol{u})$ is updated to reflect the scores based on the newly formed view pairs. The cost for each unobserved view is therefore:

$$g_u = \sum_{v \in \mathcal{V}} h(u, v) \ . \tag{2}$$

Here, $h(u, v)$ is the predicted cross entropy for classification of a view pair consisting of the unobserved view u, and the observed view v. To compute this value, we train a third CNN, denoted *CNN-3* (see Figure 3), which is identical to CNN-2 except that it is trained for regression to a cross entropy value, rather than being trained for classification to the next-best-view. CNN-3 maps a single input image to a distribution of predicted cross entropies over all viewpoints on the viewing sphere. Training pairs for this are generated by taking each training image, forming view pairs with all other views of that same object, and then computing the classification cross entropy with CNN-1. CNN-3 is then trained to minimise the L2 distance between the ground truth cross entropies and the predicted cross entropies.

We now define $\mathcal{U}^+$ as the set of 8 viewpoints adjacent to the camera's current viewpoint ($\Delta\theta = -30°$, 0, or $+30°$, $\Delta\phi = -30°$, 0, or $+30°$), representing all the positions which the camera can move to in its next step. For each viewpoint $u \in \mathcal{U}^+$, we compute the set of trajectories $\mathcal{T}_u$ over which the camera could subsequently traverse, if it were to make its next move to u. These are found by a simple undirected graph search, for a given final trajectory length. We then assign a score $s(t)$ to each trajectory $t \in \mathcal{T}_u$, by summing up the scores in the trajectory's set of unobserved views $\mathcal{U}_t$:

$$s(t) = \sum_{u \in \mathcal{U}_t} g_u \tag{3}$$

The optimum next view u^* is then chosen as the one whose best trajectory has the highest score over all of the available next views:

$$u^* = \underset{u \in \mathcal{U}^+}{\arg\max} \ \max_{t \in \mathcal{T}_u} s(t) \ . \tag{4}$$

In this way, at every step along the trajectory, the best decision is taken for the next view given the available information. As the camera follows this guided trajectory, the scores assigned to these unobserved views will change, attracting the camera towards those viewpoints which, if visited, are likely to yield a high classification confidence when processed with CNN-1.

5. Experiments

We evaluated our method on the ModelNet dataset [39], which consists of 3D CAD meshes from everyday objects over a range of scales. For our experiments, two subsets were used as in [39]: ModelNet10, containing 10 object categories with 4,905 unique objects, and ModelNet40, containing 40 object categories and 12,311 unique objects, both with a testing-training split. ModelNet is the only available dataset at this time with sufficient large-scale multi-view coverage of objects for training or testing our networks, and hence as with [39, 35], real-world experiments were not possible.

Training of the three networks was then carried out by rendering images of each model from all viewpoints on the discretised viewing sphere, and forming the full set of image pairs. Rendering was performed under perspective projection, with objects scaled uniformly to fit the viewing window, to yield images of 512-by-512 pixels for both greyscale and depth images. For rendering the greyscale images, Phong shading [30] was used, and pre-training conducted with the ImageNet $1k$ dataset [7] as with [35]. Networks trained for the ModelNet10 dataset were pre-trained on ModelNet40. During testing, unless otherwise specified, every object was tested once per viewpoint, with the trajec-

Pair Selection	Weighted ?	ModelNet10	ModelNet40
Best	No	90.1	88.2
Best	Yes	90.6	88.8
All	No	91.2	89.0
All	Yes	**91.9**	**89.5**

Table 1: Results for different implementations of our method over a sequence length of 6 views, for multi-view classification with Equation 1. The first column indicates whether all the view pairs or just the best pairs (based on λ weighting) were used for classification with CNN-1. The second column indicates whether or not the weighting λ was used, compared to an equal contribution from each view pair.

tory commencing at that viewpoint, and then the classification accuracy for that object was the average over all these trajectories.

5.1. Pairwise Recognition

First, we explored four different implementations of our multi-view recognition method, with both greyscale and depth images as input. We evaluated the performance based on two parameters: the weighting system used in Equation 1, and the way in which view pairs are formed in a view sequence. Table 1 shows recognition results for random trajectories of 6 views, with and without learned weights for each view pair, such that without the weighting, all view pairs contribute equally to the final class distribution. Then, for selection of *All* pairs, $\frac{M(M-1)}{2}$ pairs were chosen for a sequence of length M, such that every possible pair was used during recognition. For selection of *Best* pairs, the top M pairs with the greatest weight λ were chosen, such that the number of pairwise classifications was linear rather than combinatorial with the sequence length. Results show the positive effect of the weighting and the inclusion of all pairs in the sequence, with the larger number of pairs offering slightly more benefit than the inclusion of weighting. For the remaining implementations of our method, weighting was used together with the full set of view pairs.

5.2. Multi-View Recognition

We then compared our method against two recent competing methods: *ShapeNets* [39] and Multi-View CNN (*MVCNN*) [35], together with a baseline which we call *View Voting*. For ShapeNets, we used code provided by the authors, and we implemented our own version of MVCNN as per the publication details, achieving similar to their quoted results. For the View Voting method, we trained a CNN with 5 convolutional layers and 3 fully-connected layers, similar to CNN-1, to classify views independently based on the image alone. A voting system was then employed to combine the classification outputs of each view. We explored three implementations of our method: using only greyscale images, using only depth images, and then using both image modalities. With the competing methods, ShapeNets was implemented with depth images and MVCNN was implemented greyscale images, as per their original descriptions, and View Voting was implemented with greyscale images.

Table 2 shows recognition results for view sequences at an elevation of 30° from the ground plane, constrained to rotations about the gravity vector, as was the experimental setting in [39, 35]. Our method outperforms all other methods, and the combination of both greyscale and depth images achieves a small boost in performance over single image modality. As was presented in [35], rendering 2D images in a view-based manner achieves much better recognition results than the generative volumetric approach of ShapeNets. However, the capacity of ShapeNets for shape completion provides a strength that view-based methods cannot. The MVCNN method achieves the second-best results for sequences covering 360°, but for shorter sequences the performance degrades dramatically. This illustrates the unsuitability of their method for arbitrary sequence lengths, owing to testing and training requiring the same sequence length. We note that our method achieves comparable performance to MVCNN with only half the number of views. In our implementation of MVCNN, we trained on the full 360° set of images regardless of the length of the testing sequence, although this could likely be improved by training on varying sequence lengths. However, once the constraint of moving only about the gravity vector is removed, this would become intractable.

Table 3 then shows recognition results for arbitrary view sequences, where we exclude MVCNN due to its inability to generalise in this way. Our method here was provided with both greyscale and depth images. We implement each method with two variations of how the next view along the sequence is selected. *Random* chooses the next view randomly from the adjacent views, and *Straight* follows a straight path around the viewing sphere from the beginning to the end of the sequence, similar to the results in 2 except that the sequence direction is randomly selected rather than being constrained around the gravity vector. Again, our method achieves state-of-the-art recognition results, and we note that results are slightly lower than those in Table 2 due to the suitability of viewpoints at an elevation of 30° for common household objects in their canonical orientation. Choosing a straight path rather than a random path increases recognition accuracy, due to the tendency of random walks to revisit old viewpoints, or remain within a local region and hence fail to explore the object sufficiently.

		ModelNet10				ModelNet40			
Method	Image	3 views (60°)	6 views (180°)	12 views (360°)	Average	3 views (60°)	6 views (180°)	12 views (360°)	Average
View Voting	Greyscale	87.2	89.5	90.1	88.9	85.7	87.3	88.1	87.0
ShapeNets [39]	Depth	79.2	82.5	83.1	81.6	72.0	75.7	77.4	75.0
MVCNN [35]	Greyscale	84.5	89.8	92.2	88.8	82.3	88.1	89.5	86.6
Ours	Greyscale	88.5	91.4	92.8	90.9	86.2	88.8	90.7	88.6
	Depth	85.2	87.6	90.0	87.6	83.0	87.0	89.9	86.6
	Greyscale + Depth	88.8	91.9	93.2	**91.3**	87.0	89.5	91.1	**89.2**

Table 2: Recognition results over different sequence lengths, from trajectories constrained to rotations about the gravity vector, at an elevation of 30°. Numbers represent the percentage of correctly-classified objects from the test set.

		ModelNet10				ModelNet40			
Method	View Selection	3 views	6 views	12 views	Average	3 views	6 views	12 views	Average
View Voting	Random	85.5	87.2	87.6	88.3	84.0	85.8	87.0	85.6
	Straight	86.1	88.8	89.9	88.3	84.3	87.0	88.1	86.5
ShapeNets [39]	Random	76.0	81.6	82.2	79.9	70.0	74.4	77.2	73.9
	Straight	77.0	81.9	82.2	80.4	70.2	74.5	77.2	74.0
Ours	Random	86.8	87.8	91.0	88.5	86.1	88.4	90.1	88.2
	Straight	88.6	91.2	93.0	**90.9**	86.7	89.3	91.0	**89.0**

Table 3: Recognition results over different sequence lengths, from unconstrained trajectories. Numbers represent the percentage of correctly-classified objects from the test set.

5.3. Active Recognition

We then evaluated the performance of our NBV and trajectory optimisation extensions to multi-view recognition, and we compared against the NBV method of *ShapeNets* [39]. Our method here was provided with both greyscale and depth images. For both our method and ShapeNets, we implemented two strategies for view selection. *NBV Global* chooses the next-best-view from the entire viewing sphere, and *NBV Adjacent* chooses the next-best-view from the views adjacent to the current view. The *Optimised* implementation of our method is that which incorporates all images along the trajectory rather than just the start and end positions, as in Section 4.2. For all implementations, if the selected viewpoint was one which had already been visited, then the highest-scoring of all the unvisited viewpoints was selected. Table 4 shows that our method achieves the best recognition performance, and whilst the global NBV implementation sees best recognition accuracy, our *Optimised* implementation, which would be used in reality due to its greater practical efficiency, achieves a close second.

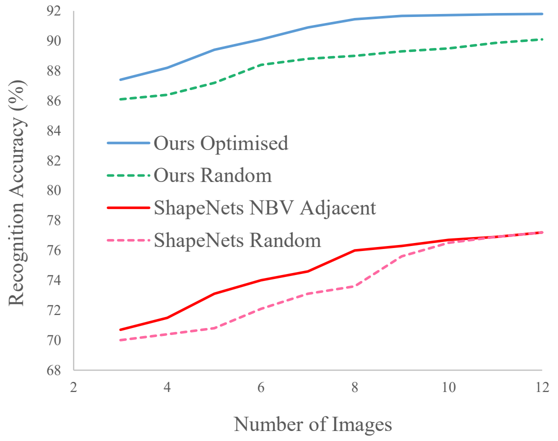

Figure 4: Recognition accuracy for different view sequence lengths on ModelNet40.

Figure 4 plots the recognition accuracy on ModelNet40 against view sequence length for our method and ShapeNets, with each under active and random trajectories and following adjacent viewpoints. Even without trajectory optimisation, our method significantly outperforms ShapeNets, and we see that our trajectory optimisation maintains an advantage over random viewpoint selection, for a range of sequence lengths. Finally, Figure 5 visualises some image sequences which our method observes under optimised trajectories. We note that the chosen trajectory often passes over the top, or beneath, the object, showing how the constraint of MVCNN to rotations around the gravity vector is sub-optimal.

Method	View Selection	ModelNet10				ModelNet40			
		3 views (60°)	6 views (180°)	12 views (360°)	Average	3 views (60°)	6 views (180°)	12 views (360°)	Average
ShapeNets [39]	NBV Global	79.2	82.0	82.9	81.3	71.1	75.6	77.2	74.6
	NBV Adjacent	78.7	81.0	82.4	80.7	70.7	74.2	77.2	74.0
Ours	NBV Global	90.4	92.8	94.0	**92.4**	88.5	91.5	92.0	**90.7**
	NBV Adjacent	88.8	91.6	93.5	91.3	87.2	89.5	91.4	89.4
	Optimised	88.9	91.9	93.9	91.6	87.4	90.1	91.8	89.8

Table 4: Recognition results over different sequence lengths, for unconstrained trajectories, using trajectory optimisation methods. Numbers represent the percentage of correctly-classified objects from the test set.

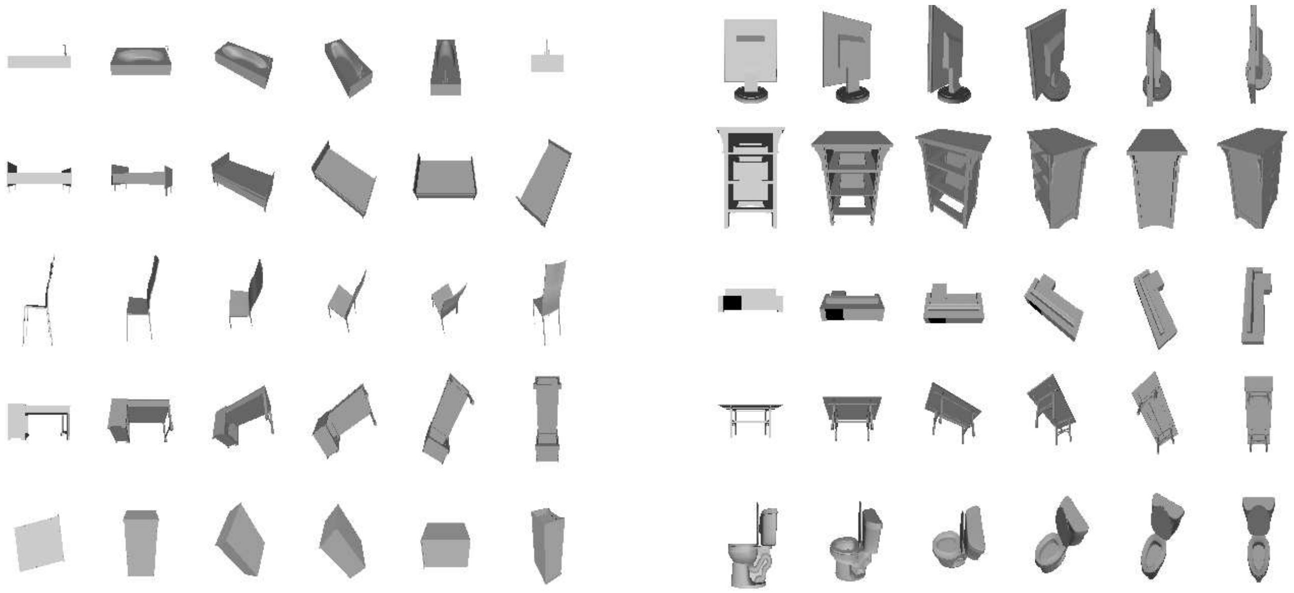

Figure 5: Example greyscale images observed from our optimised trajectories with the ModelNet10 dataset.

6. Conclusions

In this paper, we have presented a new method for multi-view object recognition over unconstrained camera trajectories, using greyscale images, depth images, or both modalities combined. We have shown that decomposing an image sequence into a set of view pairs enables training in a tractable manner for any trajectory over the viewing sphere of an object. Experiments show that our method outperforms the voxel-based generative ShapeNets method, together with the Multi-View CNN method, and we achieve state-of-the-art recognition on the ModelNet dataset. We have also shown how our pairwise method can extend to next-best-view prediction by learning discriminatively in an end-to-end manner, and this can then be incorporated into a trajectory optimisation scheme to achieve the best camera path for recognition over a given sequence length.

7. Acknowledgements

Research presented in this paper has been supported by Dyson Technology Ltd.

References

[1] S. Bai, X. Bai, Z. Zhou, Z. Zhang, and L. J. Latecki. GIFT: A Real-time and Scalable 3D Shape Search Engine. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2

[2] B. Charrow, G. Kahn, S. Patil, S. Liu, K. Goldberg, P. Abbeel, N. Michael, , and V. Kumar. Information-Theoretic Planning with Trajectory Optimization for Dense 3D Mapping. In *Proceedings of Robotics: Science and Systems (RSS)*, 2015. 3

[3] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the Devil in the Details: Delving Deep into Convolutional Nets. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2014. 4

[4] S. Chopra, R. Hadsell, and Y. LeCun. Learning a Similarity Metric Discriminatively, with Application to Face Verification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005. 3

[5] A. Collet and S. S. Srinivasa. Efficient multi-view object recognition and full pose estimation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2010. 2

[6] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005. 2

[7] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A Large-Scale Hierarchical Image Database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. 5

[8] A. Doumanoglou, T.-K. Kim, X. Zhao, and S. Malassiotis. Active Random Forests: An Application to Autonomous Unfolding of Clothes. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2014. 3

[9] R. Eigenberger and J. Scharinger. Active Perception and Scene Modeling by Planning with Probabilistic 6D Object Poses. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, 2010. 3

[10] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 32(9):1627–1645, 2010. 2

[11] R. Fergus, P. Perona, and A. Zisserman. Object Class Recognition by Unsupervised Scale-Invariant Learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2003. 2

[12] M. Firman, O. M. Aodha, S. Julier, and G. J. Brostow. Structured Prediction of Unobserved Voxels From a Single Depth Image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2

[13] S. Gupta, R. Girshick, P. Arbelaez, and J. Malik. Learning Rich Features from RGB-D Images for Object Detection and Segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2014. 2

[14] B. K. P. Horn. Extended gaussian images. In *Proceedings of the IEEE*, 1984. 2

[15] V. M. J. Lei Ba and K. Kavukcuoglu. Multiple Object Recognition with Visual Attention. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015. 3

[16] E. Johns, O. M. Aodha, and G. J. Brostow. Becoming the Expert - Interactive Multi-Class Machine Teaching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 1

[17] E. Johns and G.-Z. Yang. From Images to Scenes: Compressing an Image Cluster into a Single Scene Model for Place Recognition. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2011. 2

[18] E. Johns and G.-Z. Yang. Generative Methods for Long-Term Place Recognition in Dynamic Scenes. *International Journal of Computer Vision (IJCV)*, 106(3):297–314, 2014. 2

[19] E. Johns and G.-Z. Yang. Pairwise Probabilistic Voting: Fast Place Recognition Without RANSAC. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2014. 2

[20] M. Krainin, B. Curless, and D. Fox. Autonomous Generation of Complete 3D Object Models Using Next Best View Manipulation Planning. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2011. 3

[21] A. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet classification with deep convolutional neural networks. In *Neural Information Processing Systems (NIPS)*, 2012. 1

[22] J. Liebelt and C. Schmid. Multi-View Object Class Detection with a 3D Geometric Model. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010. 2

[23] D. Lowe. Local Feature View Clustering for 3D Object Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2001. 1, 2

[24] T. F. M. Kazhdan and S. Rusinkiewicz. Rotation invariant spherical harmonic representation of 3D shape descriptors. In *Proceedings of the Symposium of Geometry Processing*, 2003. 2

[25] T. F. M. Kazhdan and S. Rusinkiewicz. Hough transform and 3D SURF for robust three dimensional classification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2010. 2

[26] D. Maturana and S. Scherer. VoxNet: A 3D Convolutional Neural Network for Real-Time Object Recognition. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, 2015. 3

[27] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2:1115–1125, June 2005. 2

[28] B. Pepik, M. Stark, P. Gehler, and B. Schiele. Multi-View and 3D Deformable Part Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 37(11):2232–2245, 2015. 1, 2

[29] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object Retrieval with Large Vocabularies and Fast Spatial Matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007. 2

[30] B. T. Phong. Illumination for Computer Generated Pictures. *Communications of the ACM*, 18(6):311–317, 1975. 5

[31] R. Pless. Using many cameras as one. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2003. 2

[32] B. C. R. Osada, T. Funkhouser and D. Dobkin. Shape distributions. In *ACM Transactions on Graphics*, 2002. 2

[33] B. Shi, S. Bai, Z. Zhou, and X. Bai. DeepPano: Deep Panoramic Representation for 3-D Shape Recognition. *IEEE Signal Processing Letters*, 22(12):2339–2343, 2015. 2

[34] K. Simonyan and A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015. 1, 4

[35] H. Su, S. Maji, E. Kalogerakis, and E. G. Learned-Miller. Multi-view Convolutional Neural Networks for 3D Shape Recognition. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2015. 1, 2, 3, 4, 5, 6, 7

[36] A. Thomas, V. Ferrari, B. Leibe, T. Tuytelaars, B. Schiel, and L. Van Gool. Towards Multi-View Object Class Detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006. 1, 2

[37] S. Vicente, J. Carreira, L. Agapito, and J. Batista. Reconstructing PASCAL VOC. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 2

[38] K. Wu, R. Ranasinghe, and G. Dissanayake. Active Recognition and Pose Estimation of Household Objects in Clutter. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2015. 3

[39] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3D ShapeNets: A Deep Representation for Volumetric Shapes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 1, 2, 3, 4, 5, 6, 7, 8