Summary

This paper, along with the previous work by the authors of this paper, addresses the challenge of finding suitable footage for 3D reconstruction of coral amongst YouTube videos. More specifically, this paper deals with the classification of videos with camera motion that makes them suitable for 3D reconstruction of coral. The motivation behind this project lies in the use of 3D reconstruction to study and understand the state of the coral reefs around the world. Coral reefs are some of the most important and biologically diverse ecosystems on the planet. Additionally they serve many purposes to the societies close to them. Coral reefs provide coastal protection, by absorbing the energy of waves, and therefore lessening their magnitude. Coral reefs provide opportunities for tourism, and they represent an annual value of several tens of billions of dollars. Coral reefs are also important for fisheries, as around 25% of marine life depends on coral.
In previous work, we outlined and implemented a pipeline for the detection of YouTube videos suitable for 3D reconstruction of coral. This pipeline consisted of several steps. Firstly, the videos are put through a transformer model, called Video Swin Transformer, in order to detect which sections of the video consisted of undersea footage. Afterwards, the videos were run through a convolutional neural network (CNN) based model, called YOLOv8. This model would then pick out which sections of the undersea segments of the videos contained footage of coral. The last step of the pipeline involves using the same transformer model, Video Swin Transformer, to classify the camera motion of the undersea segments containing coral. We designed the pipeline this way, because in order to use footage to perform 3D reconstruction, a certain camera motion showing the entire coral colony is desired.
In order to train the models used in the pipeline, we required a dataset. To our knowledge, there previously existed no dataset suitable for our problem, so we made a new one, by taking a relevant subset of the YouTube-8M dataset and manually classifying undersea segments, coral segments and the camera motion of the coral segments.
We tested the performance of each individual step of the pipeline, and found that the performance of the camera motion classifier was lacking. That is why in this paper we sought out to improve the performance of this camera motion classifier in the previously mentioned pipeline.
Our method consists of taking the motion vector found in compressed videos, modeling these vectors using the HSI color model and therefore converting the motion vectors into color images, where the colors of the image represents the magnitude and angle of the motion vectors. Using this method we were able to model the camera motion in a video. Using these new images, we blended these images with the original footage of the YouTube video and used this to classify the camera motion in the video, using the same Video Swin Transformer model. This method was inspired by a previous paper, which did the same motion vector modeling, but instead used a CNN to classify the images.
We also implemented the previously mentioned method, which inspired our method, along with a heuristic method for results comparison. We utilized hyperparameter search to find the configuration for the CNN method and for our own transformer method. We then ran experiments on our own dataset, and compared the F1-score, precision and recall of all three methods. We found that the method using the transformer performed better on the dataset than both the heuristic method and the CNN method. Overall though, the performance of the transformer model was quite poor, meaning that this problem remains challenging.
We believe that the challenge of this problem lies in the level of noise in the data. As the data is filmed by amateurs, the typical camera motion found in these video is very

inconsistent, and that it is because of this that our method fails to properly model the motion patterns. Another challenge in this task lies in the creation of datasets. Footage that is highly suitable for coral 3D reconstruction is, in our experience, extremely rare. Therefore we had to compromise on what should be considered a positive class instance of the camera motion we are looking for, in order to gather any meaningful amount of data.

Improving camera motion classification for undersea coral videos

Sten Kirk Larsen, Lasse Enggaard Rasmussen, Dovydas Jasulaitis

ARTICLE HISTORY
Compiled May 31, 2024

ABSTRACT
The health of the planets oceans is facing a rapid decline, particularly the world's coral reefs have seen significant reduction since 2009. 3D reconstructions of coral structures are vital methods for quantifying and monitoring the health of coral reefs but such methods often require professionally obtained footage to be viable. However, there are great amounts of amateur footage available online which might be viable for use in 3D reconstruction but identifying it is a time consuming task, as coral structures require views from more than one angle. We therefore propose a model which might bridge a gap between public footage and scientific research by identifying sections of public videos which might be relevant for 3D reconstruction. In this work we present a model which identifies and isolates the desired camera motion by extracting motion vectors from video footage and converting them to HSI color images which are applied to a Swin transformer model. In order to train and validate this model we expanded upon a benchmark dataset containing data amateur footage for coral 3D reconstruction. In order to validate our model, it is tested against two other approaches. A Convolutional Neural Network (CNN) model also trained and validated upon HSI color images from vector and a Heuristic model applied to motion vectors. The CNN model and Heuristic model both performed poorly with an F1 score of 0.11 and 0.16 respectively. In contrast, Swin transformer outperformed these approaches by scoring 0.19. However, simply applying the Swin transformer without data augmentation performed the best with a score of 0.26. The HSI Swin transformer performed significantly better on the validation set, meaning the approach might be prone to over-fitting, or causes information loss for the model.

KEYWORDS
Camera motion classification, corals, HSI colors as motion, Video Swin Transformer, 3D reconstruction

1. Introduction

Our planet's oceans are facing unprecedented decline due to a combination of environmental pressures stemming from human activity. Among the most alarming aspects of this decline is the degradation of coral reefs. More than 14% of the world's coral reefs have disappeared in the period between 2009 and 2018, and the rate of decline only seems to be increasing (Souter *et al.* 2021). There are multiple causes for this decline, such as the ocean's rising temperatures and acidity levels, but our knowledge of the current and future health of the coral reefs is still not perfect. Coral reefs are one of the most biologically diverse ecosystems, and they play an important economical role for many societies via fishing, tourism, coastal protection and new biochemical compounds (Hoegh-Guldberg *et al.* 2007). Therefore, it is increasingly important to monitor reefs

so that we can detect and respond to changes caused by human activities. One method of doing this comes in the form of creating three-dimensional graphical representations of coral structures (Rossi *et al.* 2021). These models can be used to determine how live coral structures are currently faring and what happens to reef organisms when coral reefs are continuously impacted by outside disturbances (Gonzalez-Rivero *et al.* 2017).

Several conditions determine the quality of these 3D reconstructions as different reconstruction methods require different conditions. However, some of these conditions are universal. An example of such a condition is that corals must be viewed from more than one angle. This can be in the form of overlapping images taken from multiple angles, such that the entirety of a coral colony is pictured, or by filming the coral from multiple different angles (Roelfsema *et al.* 2020). Therefore, 3D reconstructions are often exclusively crafted from professionally obtained footage, which is quite limiting as such footage is expensive and in limited supply. There are vast amounts of public underwater footage readily available on the internet, and even if most of it is unusable, perhaps some of it does fulfill the conditions of 3D reconstruction. However, using amateur footage for 3D reconstruction would leave researchers with the time-consuming task of filtering through large quantities of footage before anything usable with the right conditions can be found. Therefore, there is an unmet need for a method to bridge the gap between public footage and scientific research, allowing for the vast amounts of available public data to be utilized.

A solution to this could be a method, that allows one to filter through large amounts of video data and indicate only the sections of it that might be of interest for 3D reconstruction. A way to identify only the relevant parts of the footage could be to look for footage where some of the conditions of 3D reconstruction are present. One such condition is the viewing of a coral from more than one angle.

The work presented in (Larsen *et al.* 2024), which shares several authors with this work, already explores this issue. A solution was explored in the form of a model composed of three components. An underwater detection, coral detection, and camera motion detection component. The first of the former components showed strong results, isolating only the sections of footage containing underwater coral frames. The third component was designed to detect desirable camera motion patterns to isolate footage sections where corals are viewed from multiple angles. The camera motion detection however, did not perform to expectations and could not capture the desired movement patterns of the camera. A specialized component was not applied but rather a pre-trained, general purpose neural network was used.

Thus, in this work, we explore the idea of using motion vectors in a Transformer-based model to identify camera motion for 3D coral classification. Transformer-based models have seen an increase in usage for video classification tasks. Models such as VTN (Neimark *et al.* 2021) and Video Swin Transformer (Liu *et al.* 2022) outperform state-of-the-art CNN-based models in video classification tasks. So far, we have not been able to find camera motion classification approaches that use a Transformer model. This creates an opportunity to construct a Transformer model that makes use of motion vectors to classify the camera motion in a video.

Our proposed model considers public videos sourced entirely from YouTube. The model is based on the Transformer architecture, Video Swin Transformer, and we model motion vectors as HSI colors, as the HSI color model can represent the motion vector direction and their magnitude (Pavan *et al.* 2022). We introduce mixing these HSI color images with real color to include both the motion, and picture data of the video for classification. These blended images can be fed directly to a video classification model. The noisy nature of many public amateur videos and the fact that they are underwater leads to

the usage of a transformer model approach that can consider multiple frames and weigh parts of the frames and videos differently, and an approach that makes use of both the motion data, and the frame color data.
In order to evaluate the proposed model, the model will be compared to two other methods. The first method is a heuristic method, described in (Lee and Hayes 2002), based on analyzing the motion vectors in compressed videos. The other method involves the conversion of motion vectors to HSI color images, and then classifying these images with a convolutional neural network (Pavan *et al.* 2022). The methods will be compared, by comparing their accuracy, precision and recall. The dataset used for this evaluation is made for 3D reconstruction of coral, as previously described in (Larsen *et al.* 2024), and contains data for whenever multiple angles of corals are being shown in YouTube videos. Additionally, this dataset has been extended for the purposes of this paper.

2. Related Work

Traditional approaches to camera motion classification rely on analyzing motion vectors between successive frames. Methods such as affine (Kim *et al.* 2000) and simplified affine (Gillespie and Nguyen 2004) directly analyze motion vectors that are present in an MPEG-encoded video. Lee and Hayes (2002) segment motion vectors from subsequent frames into 3×3 motion blocks, and compare these motion blocks with templates which are also defined in blocks. These approaches can be seen as heuristic approaches. Duan *et al.* (2006) propose a method to utilize mean shift clustering on block motion vectors in order to detect dominant clusters whose histogram projections are used to classify one of six camera motion types for video indexing. Hasan *et al.* (2014) take this further to classify cinematographic shots, and introduce more steps such as motion consistency analysis to remove noisy motion vectors. (Duan *et al.* 2006) and (Hasan *et al.* 2014) make use of support vector machines to perform the classification.
Determining motion vectors can be done through block matching techniques, optical flow approaches, or tracking points of interest (Rublee *et al.* 2011). Bommes *et al.* (2020) provide an approach to extract motion vectors from H.264 or MPEG-4 encoded compressed videos. This works by making use of the compression technique which, instead of saving full-color frames for every frame, saves key-frames. Intermediate non-key frames get segmented into macro-blocks that are encoded by the motion vectors, which define the movement to past or future frames. This way, intermediate non-key-frames reuse information from key-frames.
Heuristic approaches limit the reliability and generalization of camera motion classification (Ouenniche *et al.* 2021). These limitations have led to the exploration of machine-learning approaches for camera motion classification. Convolutional neural networks (CNNs) see usage within image and video classification domains. Tran *et al.* (2015) are among the first to propose a 3D CNN architecture for video classification. For camera motion classification, Ouenniche *et al.* (2021) propose a 3D CNN architecture based on ResNet (He *et al.* 2015) that accepts a video as an input and outputs the type of motion. This approach is trained and tested on 2-second-long videos that were collected from various YouTube videos and different movies and achieves an average accuracy of 94% on a dataset generated by Ouenniche *et al.* (2021). Pavan *et al.* (2022) propose an approach that works with motion vectors extracted from a H.264 encoded video, modeling the underlying motion vectors in the HSI color model, converting HSI colors to RGB colors, and feeding the converted RGB images to a 2D CNN, classifying 11 different camera motion types per frame. Pavan *et al.* (2022) achieve an accuracy of

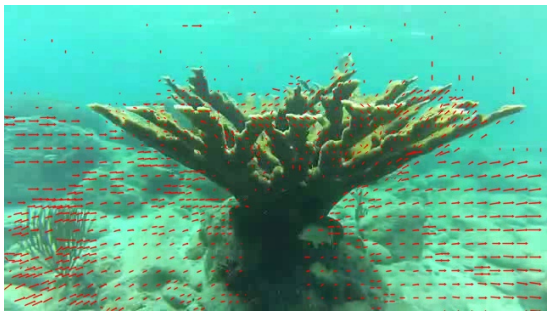
98% on H.264 encoded video sequences. However, the generalizability of these accuracy results is questionable as the dataset consists of 26 curated high-quality videos in which the camera motion is deliberate and clean.
To the best of our knowledge, none of these models have been attempted on a challenging task such as the one presented by Larsen *et al.* (2024) where it is attempted to detect subtle camera motion patterns in footage with a large degree of noise. Here, we suggest a Transformer-based approach where we make use of modeling motion as HSI color images, comparing it to the CNN based approach that does this (Pavan *et al.* 2022), and a heuristic approach (Lee and Hayes 2002).

3. Augmenting video data with motion as colors

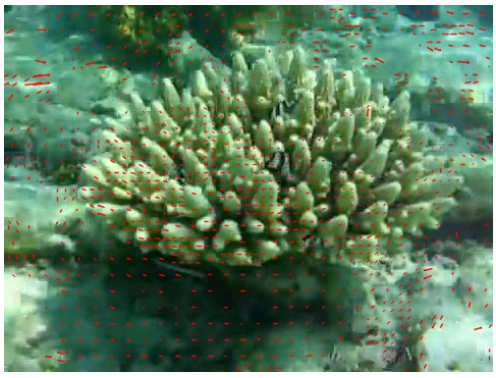
In this section, we describe the process of modeling motion vectors in videos as HSI color images, and the process of mixing these images with real frame colors. However, first we want to clarify what the motion we want to classify looks like in a 2D motion vector field. This is necessary both to properly describe what we are classifying in videos and to make it possible to formulate a heuristic for model comparison in Section 5.

3.1. *Multiple angle view as a vector field*

We review the block motion vector fields in a few videos from the benchmark proposed in (Larsen *et al.* 2024). We find that there are two general situations where the camera views corals from multiple angles; one where the camera pivots around the coral, and one where the camera passes the coral in a way that shows multiple, small angles of it, i.e briefly passes the coral over the top. These types of motion are shown in figures 1 and 2 respectively. Notably, the pivoting motion generally has less movement towards the center of the frame, and the passing over the top motion has movement that is in the same direction across the whole frame, but the motion is more pronounced on the corals as they are closer to the camera than the rest.

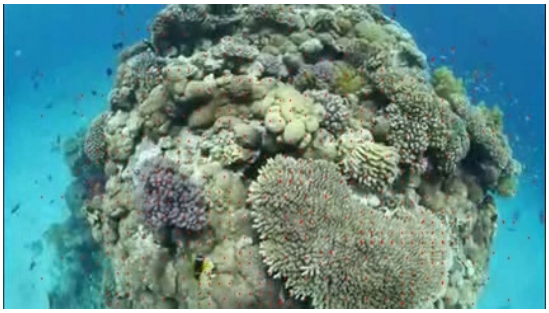


(a) First example of pivoting around a coral. This one has clear motion in one direction, except for the middle where the coral is and is appearing stationary in the video.

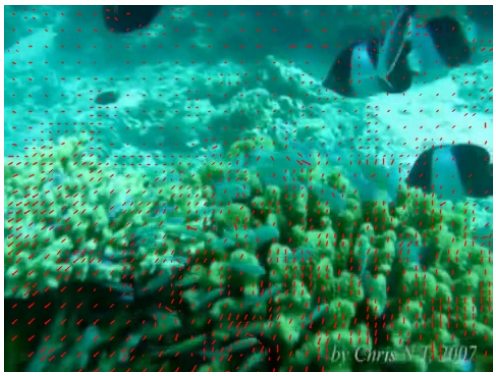


(b) Second example of pivoting around a coral, camera is shaky, so the vectors are a less uniform, but this one rotates around it from the top, so the vectors do not share the same direction around the coral.

Figure 1. Examples of motion when camera pivots and rotates around corals.



(a) First example of passing over the top of some corals. The vectors are mostly pointing downwards because the camera is passing over the corals from bottom to top. Motion vectors on the left and right edge of the frame are very small.



(b) Second example of passing over the top of a coral. The vectors are pointing downwards. Here, the vectors on the corals have a higher magnitude than those in the upper half of the frame because the corals are closer to the camera.

Figure 2. Examples of motion when camera passes over the top of some corals.

3.2. *Motion vectors to HSI color images*

The first process in our approach is to extract motion vectors from compressed videos. The ideal approach here is to utilize the block motion vectors already present in H.264 encoded videos, which is all videos that are downloaded in the MP4 format from YouTube. Bommes *et al.* (2020) provide a tool to extract block motion vectors for each frame in a video. The idea that Pavan *et al.* (2022) introduce is modeling block model motion vector orientation and magnitude as HSI colors, where the orientation is the Hue, the magnitude is the Saturation, and the Intensity is fixed at 100%. This HSI representation of the block motion vectors can then be converted to a RGB representation, which can be fed into a video classification network. This representation is depicted in Figure 3. It is important to note that when using frames that are segmented into blocks, the frame size decreases based on the number of blocks. The block size depends on the encoding. In MP4 videos downloaded from YouTube the block size varies—8×8, 8×16, 16×8, 16×16—within the same frame, depending on the detail required for compression, which makes the resolution inconsistent. Furthermore, we invert motion vectors in frames that reference a future frame, as these vectors are otherwise inverted, creating unnecessary noise for classification.

The Hue and Saturation parts of the HSI representation can be calculated using the orientation and the magnitude of the motion vectors, respectively. A motion vector consists of two components, the horizontal and the vertical component. Therefore, let $MV = (MV^X, MV^Y)$, in which MV^X is the horizontal component, and MV^Y is the vertical component. From this, we can calculate the orientation MV_{ori} and magnitude MV_{mag} using the following formulas:

$$MV_{ori} = arctan(\frac{MV^Y}{MV^X}), 0 \leq MV_{ori} \leq 2\pi \tag{1}$$

$$MV_{mag} = \sqrt{(MV^X)^2 + (MV^Y)^2} \tag{2}$$

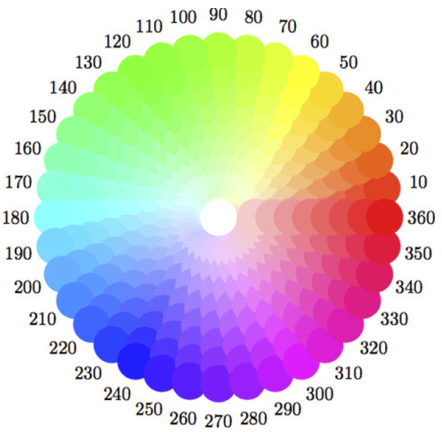


Figure 3. HSI color gamut depicting the possible colors given an angle. The magnitude of the vectors can be interpreted as the distance from the center of the color gamut, or saturation of the color. Figure is from (Pavan *et al.* 2022).

Using MV_{ori} and MV_{mag}, the HSI color for a block can be assigned as:

$$Hue \leftarrow MV_{ori}, Saturation \leftarrow MV_{mag}, Intensity \leftarrow 100\%$$

This assignment corresponds to converting the two-dimensional coordinates of motion vectors into an RGB image that purely represents motion in the video. By structuring motion input as images, we enable image/video classification models to classify motion in videos. However, using motion vectors from H.264 encoding includes getting motion from moving objects, and moving point of view. Therefore, using motion vectors from a video is not entirely representing the motion of the camera. Figure 4a shows how an image generated from motion vectors looks.

3.3. *Mixing HSI color image with real frame*

An approach that considers multiple frames that only consist of motion vectors converted to HSI color images has the possibility of being completely unable to classify anything. This is due to the inherent noise in motion vectors extracted from compressed underwater videos. To mitigate this, we mix the HSI color image with real colors from the frame. This approach also enables the model to recognize the presence of corals in the frames while considering motion. The mixing is done with a simple linear interpolation between the HSI color image and the real image, with a mixing coefficient to decide how much the resulting image resembles one or the other. In our implementation, we set the mixing coefficient to a fixed value of 0.6, but it would be interesting to have it as a hyperparameter. The resulting image looks like the real frame with the HSI colors as an overlay.

4. Video Swin Transformer

This section describes the architecture of the Video Swin Transformer, which we use as our classification model. We choose Video Swin Transformer because it is a well known video classification model that can consider motion by employing techniques such as 3D shifted windows. We use the tiny version of Video Swin Transformer (Swin-

(a) Motion vectors from Figure 1a converted to an HSI representation, and then to an RGB image. Following Figure 3, most of the camera movement is to the right, and the middle of the frame, where the coral is, has less camera movement.

(b) HSI representation of motion vectors from Figure 1a converted to an RGB image and mixed with the real frame. Here, the mixing coefficient is 0.5 to make it clear.

Figure 4. Motion vectors and frame from Figure 1a represented as HSI colors, and HSI colors mixed with the real frame.

T) proposed by (Liu *et al.* 2022). Video Swin Transformer is a development of Swin Transformer (Liu *et al.* 2021), so it follows similar model architecture with significant modifications that allow it to perform video recognition. Figure 5 shows an overview of the architecture for the tiny version of Video Swin Transformer. The input undergoes a 3D patch partitioning process, is linearly embedded into a feature space of size C, and passes through four stages where multi-head self-attention is calculated and the dimensions are downscaled by a factor of 2. The input video has the shape $T \times H \times W \times 3$, where T is the temporal dimension, H is the height of the frames, W is the width of the frames, and 3 represents the three color channels for RGB.

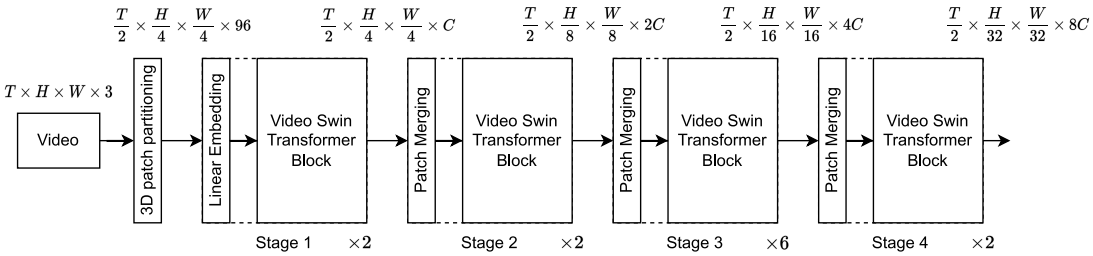


Figure 5. Video Swin Tiny architecture overview, based on (Liu *et al.* 2022).

3D patch partitioning

The first step of the model is partitioning the input into 3D patches, and linearly embedding the three color channels into 96 features. The result of this has the shape $\frac{T}{2} \times \frac{W}{4} \times \frac{H}{4} \times 96$. Then, the 96 features are linearly embedded into C amount of features. C varies depending on the model size. In Swin-T it is 96.

Multi-head self-attention

Figure 5 shows each Video Swin Transformer block where multi-head self-attention (MSA) is performed. MSA is chosen because a global self-attention representation would be too expensive to compute. MSA is performed within each 3D window. The 3D windows are evenly partitioned patches along the temporal dimension, the height, and

width, in a way that ensures the windows do not overlap. Each window has the size $P \times M \times M$, which means the partitioning yields $\frac{T'}{P} \times \frac{H'}{M} \times \frac{W'}{M}$ windows.

3D shifted windows

Due to the application of multi-head self-attention within non-overlapping windows, connections between the windows are absent. To address this, a 3D window shifting process is introduced after the initial window partitioning of the first layer in each stage. The windows are shifted by $(\frac{P}{2}, \frac{M}{2}, \frac{M}{2})$ patches, and computing multi-head self-attention for these windows introduces connections to the previous layer's windows. Furthermore, a relative position bias is introduced to include bias to each head in the self-attention computation. This positional bias B is a parameterized variable in the self-attention computation which is sampled from matrix $\hat{B} \in \mathbb{R}^{(2P-1)\times(2M-1)\times(2M-1)}$.

$$Attention(Q, K, V) = SoftMax(QK^T/\sqrt{d} + B)V \tag{3}$$

5. Experimental evaluation

In this section, we present the performance of the proposed approach described in Sections 3 and 4, on the extended multi-view coral benchmark described in Section 5.1. We compare our approach to others: a CNN based a approach which makes use of motion vectors represented in HSI colors (Pavan *et al.* 2022), and a heuristic approach (Lee and Hayes 2002). All the models are trained and evaluated on the same benchmark. Pre-processing is done for each video before training and testing. For the deep-learning approaches, each video is converted into an HSI color representation of the motion vectors. Furthermore, each video is divided into 10-second segments, each labeled as either containing (1) or not containing (0) corals from multiple angles. As the videos do not exclusively contain coral footage, we only consider segments that include corals. This is a choice made because the benchmark assumes that the segments considered for a multiple-angle view of coral already pass the filter, which filters out footage that is not underwater or does not contain corals. Additionally, each frame is resized to a 122×122 matrix and normalized using the mean and standard deviation of the RGB values. The tensors are arranged as $B \times C \times T \times W \times H$, representing the mini-batch size (B), color dimension (C), temporal dimension (T), frame width (W), and frame height (H). From the videos, we use 70% for training, 20% for testing, and 10% for validation which means we do not split on the 10-second segments. Before evaluating performance on the test set, we use the training and validation sets to conduct a hyperparameter search with Weights & Biases (Biewald 2020). We use AdamW (Loshchilov and Hutter 2019) as the optimizer for training, cosine decay learning rate scheduler, and binary cross-entropy loss as the loss function. We present the results using the F1 score, recall, and precision metrics. For training, we instantiate the Video Swin Transformer on pre-trained weights from a model trained on the Kinetics400 dataset (Liu *et al.* 2022). Then, we replace the last layer with a two-layer linear network that maps the internal features of the layer before the last layer to one class. Additionally, we employ random weighted sampling to under-sample the negative class during training to address the imbalance in the training set.

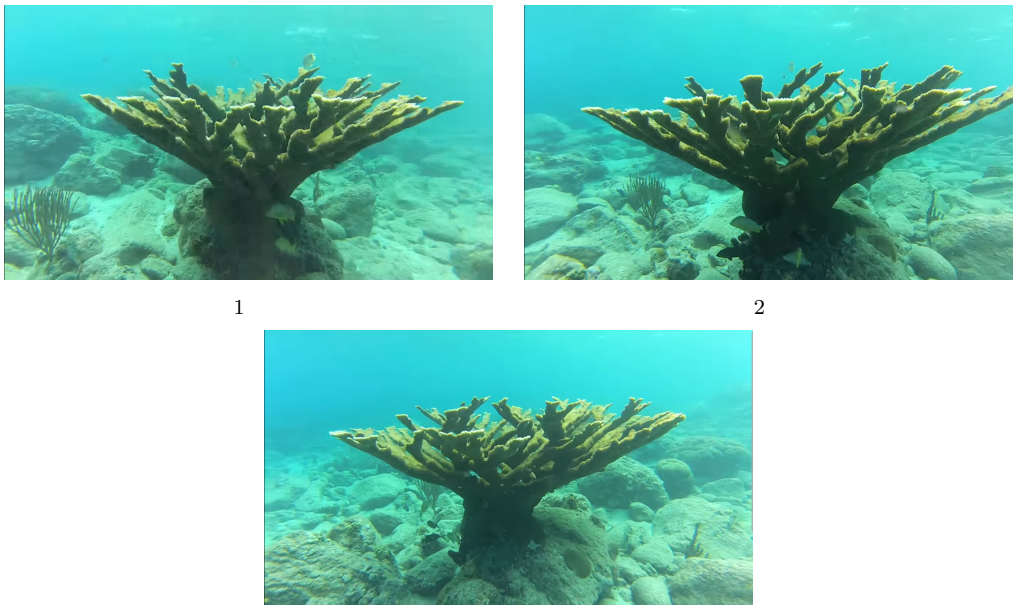
5.1. *Benchmark*

Here we describe the benchmark created in (Larsen *et al.* 2024) and extended for the purposes of this paper. This benchmark provides the training, testing, and validation data for the experiments.
As previously there existed, to our knowledge, no datasets/benchmarks containing data for 3D reconstruction of coral, one had to be created. As outlined in (Larsen *et al.* 2024), we created a dataset from YouTube videos by taking a subset of the YouTube-8M dataset (Abu-El-Haija *et al.* 2016) tagged as containing underwater footage and corals. These videos were then classified by writing down which segments contained undersea footage, which segments contained corals, and which segments viewed the same corals from multiple angles. For the purposes of this paper, the coral camera motion data will be used for the training and validation of multiple camera motion models. Additionally, the dataset was extended for this paper by classifying the camera motion in an additional 122 videos containing coral footage, making it a total of 354. Notably, 38 of the additional videos contain segments with corals viewed from multiple angles. These videos were retrieved by running an implementation of the first two stages in the pipeline described by Larsen *et al.* (2024), and were manually labeled. After segmenting all of the videos into 10-second segments, the total number of positive segments is 414, and the total number of negative segments is 3077.

One challenge encountered in creating this benchmark was the overall quality of the camera motion data found in the YouTube videos. Ideally, for coral 3D reconstruction, the camera motion consists of a full 360° pivot around the coral, showing the entire coral structure (Ferrari *et al.* 2017). However, in our experience, this kind of footage very rarely occurs, and in 342 videos containing footage of coral, we have found no instance of what would be considered ideal footage for coral 3D reconstruction. This means we had to soften the requirement of the camera motion such that a positive instance of the camera motion would simply be an instance where the camera pivots around the coral and shows multiple angles, regardless of how many degrees the camera pivots or how much is shown of the coral structure.
Despite this, instances of quite high-quality camera motion were found. Figure 6 shows an example of some of the highest-quality footage we found in making the benchmark.

Heuristic approach

A heuristic approach to classification of camera motion is used as a point of comparison to the proposed method. This heuristic method is based on (Lee and Hayes 2002). This method is based on looking at averages of the motion vectors in MPEG-compressed videos and then comparing these averages to templates to match the motion to a certain template. Given that the proposed model described in (Lee and Hayes 2002) is to classify basic camera operations, some modifications were made so as to suit the purpose of identifying the desired camera motion of this work better.
The first step of the approach is filtering of noisy motion vectors. This is done by applying a simple median filter to the magnitude of the horizontal and vertical components of the motion vectors separately.
In the next step, the motion vector field (MVF) of the frames of the video is divided into 9 regions called sub-MFVs. These sub-MVFs are then categorized as either background sub-MVFs or object sub-MVFs. The purpose of this is to decide which sub-MVFs are most relevant to consider when attempting to classify the camera motion, with the sub-



1 2

3

Figure 6. Pictures showing some of the best camera motion around a coral found in the YouTube videos. Ideally, the camera motion consists of a pivot around the coral structure, showing as much of the structure as possible.

MVFs categorized as background being the only ones considered. Then, within each background sub-MVF, the average magnitude and angle of the macroblock vectors are found. Originally, this method calculated the average magnitude of the movement within background sub-MVFs and compared it to some preset threshold whenever a camera movement occurs, so that only motion vectors of a significant magnitude were considered. However, given that the desired motion vectors this model is searching for are both small and large, then the aspect of the model which considers magnitude became unusable and was thus discarded. Despite this, every frame was still grouped into a segment of a given size, which are used to create phase histograms to better capture the angle distributions over multiple frames by aggregating them. These histograms are then compared to histograms created based on a number of different motion templates.

Motion templates as seen in Figure 7 describes a certain camera motion, for example,

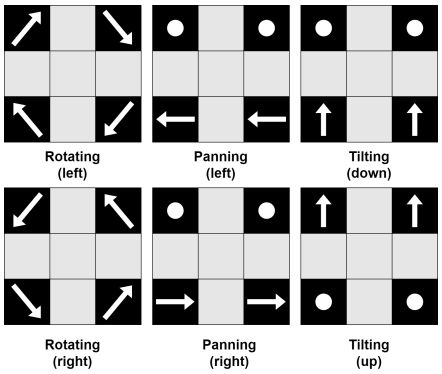


Figure 7. Examples of modified camera motion templates as opposed to those presented in (Lee and Hayes 2002)

the camera panning to the right, would make the background sub-MVFs point to the left (or have an angle of 180°), so a template for panning to the right would consist of background sub-MVF vector pointing to the left. The original templates described in (Lee and Hayes 2002) were originally created to capture only basic camera movements and have therefore been modified to better reflect the desired camera motion which this model attempts to identify. The template that best describes the vector angles of the background sub-MVFs of the actual frame segment is found by comparing their phase histograms. If the match is close enough the frame is classified as belonging to the camera motion described by the template. If enough consecutive segments are classified as having the same type of camera motion, that type of camera motion is registered as detected within a time frame.

2D CNN approach

One of the approaches we compare to is the one proposed by Pavan *et al.* (2022). Pavan *et al.* (2022) build a small two-dimensional CNN that recognizes camera motion from motion vectors converted to HSI color images in a highly curated dataset of videos. Notably, this approach uses HSI color images only, the images are not mixed with the real frame. Furthermore, the input data is of size 44×34 with the tensor shape $B \times C \times W \times H$, which means the CNN only considers one frame at a time at a much smaller size. Here, we train this CNN on our dataset and compare it with our approach, where the frames correspond to the same segments in the training, test, and validation sets.

Hyperparameter search

In this section, we describe the hyperparameter search for the attention model, and the 2D CNN approach. For the attention model, we have the following training hyperparameters: mini-batch size, epochs, learning rate, weight decay, and warm-up epochs. We use Weights & Biases (W&B) (Biewald 2020) for hyperparameter search by setting up possible value ranges for each hyperparameter, and trying to maximize the F1 score on the validation set. We set up W&B to perform hyperparameter search by using Bayesian optimization, and allowing it 30 attempts to try different hyperparameter configurations. The models from this hyperparameter search that perform the best on the validation set, are tested on the test set.

Results

We carry out experiments for our approach, the CNN approach, and the template matching heuristic approach. The results are shown in Table 1.

Overall, our approach marginally outperforms the CNN and the heuristic approaches in the combined F1 score. However, according to the recall score our approach only classifies 26% of the positives as positives, but it does not yield as many false positives as the CNN and the heuristic which leads to a higher precision score.

Discussion

Our tests show that our benchmark remains challenging, as the baseline approaches perform much worse than the original papers suggested. We find that the HSI-CNN

Table 1. Results of running each model on the test set. HSI-Swin-T is our approach of mixing HSI images generated from motion data with real frames, and classifying with the Video Swin-T model. For the Swin-T, the input comprises just the video segments.

	F1 score	Recall	Precision
HSI-CNN	0.11	0.53	0.06
Template matching	0.16	0.37	0.11
Swin-T	0.26	0.51	0.17
HSI-Swin-T	0.19	0.26	0.15

approach is not able to capture anything, and is very close in performance as simply randomly assigning positive or negative to each class. For instance, the amount of true positives and false negatives was almost the same. This is also reflected in the recall score, as 53% of the positive entries in the test set are classified correctly, but the precision is only 0.06. Given the class imbalance for frames (444 positives versus 7684 negatives), randomly guessing the label for each frame would yield a precision of 0.054. This means the CNN was only marginally better than just guessing the label randomly.

The template matching heuristic approach has a precision score higher than HSI-CNN, indicating that the positive class classification is slightly more reliable with less false positives. However, recall is lower, indicating that a significant amount of the positive class was falsely classified as negative. This is expected as the noisy nature of the videos makes it almost impossible to generalize motion vectors in a way that does not get a lot of false negatives or false positives. Especially given that the model was originally designed to capture macro movements and it had to be modified for this work so as to also capture more subtle movement patterns. However, it is still interesting that a simple approach like this outperforms the CNN approach.

Our approach of mixing HSI images with real frames seems to outperform both the template matching heuristic approach, and the HSI-CNN approach. However, simply feeding the real frames to a Swin-T model provides results that also outperform these approaches in terms of F1 score, while having a reasonable recall without sacrificing as much performance in precision. Our arbitrary choice of mixing coefficient (0.6) for the HSI-Swin-T model has lead to a worse result than the base Swin-T model. We could not test the mixing coefficient as a hyperparameter, as trying a different mixing coefficients means having to wait for video preprocessing each time, and takes more time than training the model 30 times. Further evaluation should be done to conclude if using HSI colors to represent motion in these frames either adds noise to the data, or if a mixing coefficient can be found that helps the model to classify the motion. Notably, Swin-T model on its own implicitly analyzes motion by performing 3D shifted windows, so mixing the frames with HSI colors could be detrimental to the model performance. Furthermore, our approach had a higher F1 score, recall and precision than the base Swin-T on the validation set. That could potentially mean that the sweeps found an optimal hyperparameter configuration to classify the validation set. This did not transfer to the test set, which would usually indicate a case of over-fitting. We expect that mitigating noise in motion vectors, making the motion vectors more consistent throughout the segment would assist the classification model in classifying the camera motion. However, we show that our approach (HSI-Swin-T) and the base Swin-T approach outperform the previous camera motion classification approaches, and there is room for improvement in this particular task.

6. Future Work

In this section, we describe possible ways to improve our approach. An obvious development would be to implement a method such as a median filter to eliminate noisy motion vectors or approximate the general motion in a segment. This would help a classification model in learning and testing as the motion would be less ambiguous throughout a segment.

As multi-modal approaches have seen success within video classification domains, one could structure the attention model with a multi-modal approach. This could be done by adding a multi-modal fusion layer at the beginning of the model, resulting in the motion vectors being a separate input from the frames. Additionally, generating synthetic data could help including less noisy data for the training process, and could mitigate the class imbalance. For instance, one could use a virtual 3D setting (Qiu *et al.* 2017) to generate scenes with camera motion that would allow for 3D reconstruction. Furthermore, one could extend the benchmark and model to include not just generic desired camera motion but more specific types of camera motion. Hence allowing the model to produce a more specific result should a more precise type of movement be desired.

7. Conclusion

The goal of this work was to create a model which could be used to identify camera motion relevant for 3D reconstruction of corals in amateur footage. We extend the multi-view coral benchmark to include more videos of corals viewed from multiple angles. We employ a Video Swin Transformer based model and utilize HSI color images created from motion vectors to augment video data for classification. We compare this model to two other camera motion classification approaches, a CNN model and a heuristic model which both also utilize motion vectors in compressed videos. We find that the transformer based video classification model, i.e., the Video Swin Transformer, outperforms both of the original camera motion classification approaches with or without HSI color images created from motion vectors. However, in our testing, data augmentation via converting motion vectors to HSI colors is prone to introduce noise, or cause the model to start over-fitting for the validation set, causing it to perform worse than only performing normalization and rescaling on the input data. Overall, the problem remains challenging, and more work could be done to improve camera motion classification models in amateur underwater videos.

References

Abu-El-Haija, S., *et al.*, 2016. Youtube-8m: A large-scale video classification benchmark. *arXiv preprint, arXiv:1609.08675*.

Biewald, L., 2020. Experiment tracking with weights and biases. Available from: https://www.wandb.com/.

Bommes, L., Lin, X., and Zhou, J., 2020. Mvmed: Fast multi-object tracking in the compressed domain. *In*: *2020 15th IEEE Conference on Industrial Electronics and Applications (ICIEA)*. 1419–1424.

Duan, L.Y., *et al.*, 2006. Nonparametric motion characterization for robust classification of camera motion patterns. *IEEE Transactions on Multimedia*, 8 (2), 323–340.

Ferrari, R., *et al.*, 2017. 3d photogrammetry quantifies growth and external erosion of individual coral colonies and skeletons. *Scientific Reports*, 7 (1), 16737. Available from: https://doi.org/10.1038/s41598-017-16408-z.

Gillespie, W. and Nguyen, D., 2004. Robust estimation of camera motion in mpeg domain. *In*: *2004 IEEE Region 10 Conference TENCON 2004.* vol. A, 395–398 Vol. 1.

Gonzalez-Rivero, M., *et al.*, 2017. Linking fishes to multiple metrics of coral reef structural complexity using three-dimensional technology. *Scientific Reports*, 7, 13965.

Hasan, M.A., *et al.*, 2014. Camhid: Camera motion histogram descriptor and its application to cinematographic shot classification. *IEEE Transactions on Circuits and Systems for Video Technology*, 24 (10), 1682–1695.

He, K., *et al.*, 2015. Deep residual learning for image recognition. *arXiv preprint, arXiv:1512.03385.*

Hoegh-Guldberg, O., *et al.*, 2007. Coral reefs under rapid climate change and ocean acidification. *Science*, 318 (5857), 1737–1742. Available from: http://www.jstor.org/stable/20051804.

Kim, J.G., *et al.*, 2000. Efficient camera motion characterization for mpeg video indexing. *In*: *2000 IEEE International Conference on Multimedia and Expo. ICME2000. Proceedings. Latest Advances in the Fast Changing World of Multimedia (Cat. No.00TH8532).* vol. 2, 1171–1174 vol.2.

Larsen, S.K., Jasulaitis, D., and Rasmussen, L.E., 2024. A benchmark and pipeline for identification of videos for coral 3d reconstruction. *Aalborg Universitet.*

Lee, S. and Hayes, M.H., 2002. Real-time camera motion classification for content-based indexing and retrieval using templates. *In*: *2002 IEEE International Conference on Acoustics, Speech, and Signal Processing.* vol. 4, IV–3664–IV–3667.

Liu, Z., *et al.*, 2021. Swin transformer: Hierarchical vision transformer using shifted windows. *In*: *2021 IEEE/CVF International Conference on Computer Vision (ICCV).* 9992–10002.

Liu, Z., *et al.*, 2022. Video swin transformer. *In*: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).* 3192–3201.

Loshchilov, I. and Hutter, F., 2019. Decoupled weight decay regularization. *arXiv preprint, arXiv:1711.05101.*

Neimark, D., *et al.*, 2021. Video transformer network. *arXiv preprint, arXiv:2102.00719.*

Ouenniche, K., Tapu, R., and Zaharia, T., 2021. A deep learning-based approach for camera motion classification. *In*: *2021 9th European Workshop on Visual Information Processing (EUVIP).* 1–6.

Pavan, S., Kolanu, H., and Okade, M., 2022. Cnn-based camera motion classification using hsi color model for compressed videos. *Signal, Image and Video Processing*, 16, 1–8.

Qiu, W., *et al.*, 2017. Unrealcv: Virtual worlds for computer vision. *In*: *25th ACM international conference on Multimedia*, MM '17, New York, NY, USA. Association for Computing Machinery, 1221–1224.

Roelfsema, C., *et al.*, 2020. A protocol for extracting structural metrics from 3d reconstructions of corals. *Frontiers in Marine Science*, 7, 470.

Rossi, P., *et al.*, 2021. Needs and gaps in optical underwater technologies and methods for the investigation of marine animal forest 3d-structural complexity. *Frontiers in Marine Science*, 8. Available from: https://www.frontiersin.org/articles/10.3389/fmars.2021.591292.

Rublee, E., *et al.*, 2011. Orb: An efficient alternative to sift or surf. *In*: *2011 International Conference on Computer Vision.* 2564–2571.

Souter, D., *et al.*, eds., 2021. *Status of coral reefs of the world: 2020.* International Coral Reef Initiative (ICRI): Global Coral Reef Monitoring Network (GCRMN) and International Coral Reef Initiative (ICRI).

Tran, D., *et al.*, 2015. Learning spatiotemporal features with 3d convolutional networks. *In*: *2015 IEEE International Conference on Computer Vision (ICCV).* 4489–4497.