A Transformer-Based Network for Dynamic Hand Gesture Recognition

Andrea D’Eusanio¹, Alessandro Simoni¹, Stefano Pini¹, Guido Borghi², Roberto Vezzani¹, Rita Cucchiara¹
¹Department of Engineering “Enzo Ferrari”, University of Modena and Reggio Emilia (Italy)
²Department of Computer Science and Engineering, University of Bologna (Italy)
{s.pini, name.surname}@unimore.it, guido.borghi@unibo.it

Abstract

Transformer-based neural networks represent a successful self-attention mechanism that achieves state-of-the-art results in language understanding and sequence modeling. However, their application to visual data and, in particular, to the dynamic hand gesture recognition task has not yet been deeply investigated. In this paper, we propose a transformer-based architecture for the dynamic hand gesture recognition task. We show that the employment of a single active depth sensor, specifically the usage of depth maps and the surface normals estimated from them, achieves state-of-the-art results, overcoming all the methods available in the literature on two automotive datasets, namely NVidia Dynamic Hand Gesture and Briareo. Moreover, we test the method with other data types available with common RGB-D devices, such as infrared and color data. We also assess the performance in terms of inference time and number of parameters, showing that the proposed framework is suitable for an online in-car infotainment system.

1. Introduction

The recent introduction of affordable RGB-D devices, which couple RGB cameras with active depth sensors, has attracted the interest of the research community in Natural User Interfaces (NUIs), in which the interaction is conveyed through the body of the user [39, 30] instead of traditional tools, like keyboards and mouse. In this context, the ability to recognize dynamic hand gestures, i.e. a combination of static hand poses and motion, without the use of contact-based sensors is an enabling and crucial task. The hand gesture recognition task is commonly tackled through the use of RNNs [22, 29], such as LSTMs [50, 7], architectures that are able to model the temporal and sequential nature of dynamic gestures. Alternatively, authors have proposed to classify temporal sequences using 3D CNNs [51, 31], standard CNNs [15, 14] or other machine learning methods, like HMMs [28, 6] or HOG and SVM [38, 18].

The recent spread of attentive models, which are characterized by the use of self-attention mechanisms, has come with the introduction of new approaches, such as the Transformer [43], which can replace traditional recurrent modules, such as RNNs and LSTMs. However, these approaches have not yet been deeply explored for the analysis of visual data and, in particular, for the dynamic hand gesture recognition task.

In this paper, we propose a method to classify dynamic hand gestures based on the Transformer architecture, which was originally developed for the machine translation and language modeling tasks. We propose the use of RGB-D or active depth devices and, in particular, we show that the use of depth maps and the surface normals estimated from them leads to state-of-the-art results. In addition, we investigate the adoption of the other data streams usually provided by RGB-D sensors, i.e. infrared amplitude and color images, and derived data, such as optical flow.

The employment of light-invariant data sources – depth and infrared images – guarantees the applicability of the proposed method for a Human-Computer Interaction (HCI) system able to work even in presence of dramatic and fast light changes, as often occurs in the automotive setting [37]. Indeed, the presence of tunnels and trees or bad weather conditions can strongly influence the quality of the acquired data in this scenario. Moreover, the use of inexpensive and compact cameras, which can be easily integrated in the car cockpit, is an optimal choice in order to avoid obstructions to the driver’s movements or gaze. It is shown [46, 13] that the presence of a NUI-based system for the interaction with the infotainment system of a car can significantly reduce the driver’s manual and visual distraction [4, 5] often responsible for fatal road crashes.

For these reasons, the choice of datasets to test the proposed system is automotive-driven: we exploit two publicly released datasets, namely NVidia Dynamic Hand Gesture [33] and Briareo [31]. They are both acquired in a realistic car simulator through several acquisition sensors placed in different position inside the car cockpit, as detailed in Section 4.1. When tested on these datasets, the
The proposed transformer-based architecture achieves state-of-the-art results, overcoming existing literature competitors. Moreover, the proposed method is flexible, since it can be adapted to the available data types and is able to run in real-time on a dedicated graphics card. The proposed architecture is implemented in PyTorch 1.5 and the code is available online.

2. Related Work

In the literature, the hand gesture recognition task has been approached using different strategies which enable the temporal observation of an action performed by a human. However, recent architectures [42, 33, 8], which exploit the potential of 3D Convolution in extracting temporal features from videos, become milestones as an action recognition system.

As many tasks in the computer vision field, the hand gesture recognition task can rely on different types and combination of input data. Therefore, from a general point of view, methods available in the literature can be grouped as unimodal and multimodal.

In the unimodal case a single input (e.g. RGB, infrared, depth) is used at a time. Köpükülü et al. [25] adapt state-of-the-art architectures, i.e. C3D [42] and ResNet [21], in a lightweight framework composed of a detector, that detects the beginning and the end of a gesture, and the gesture classifier. Since 3D CNNs needs more training data due to the larger number of parameters with respect to 2D CNNs, the networks are pre-trained on one of the largest public hand gesture dataset, namely Jester [32], and then fine-tuned on other datasets. In [10] authors exploit 3D hand joints to reconstruct the hand skeleton and then perform the gesture classification capturing the motion and the hand shape through a video sequence. Unfortunately, their method gets quite low results on datasets without high-quality hand skeleton annotations. Finally, with the recent success of self-attention [43] in emulating the human visual perception, an attention-based network has been introduced by Dhingra et al. [12]. They use a 3D CNN model in which 3 attention blocks are positioned between the residual modules in order to learn features at different scales. Since they train their network from scratch, they obtain good results only on datasets with a large amount of training data.

In the multimodal setting two or more input types are exploited for the recognition task. In [35] authors propose a novel architecture that, exploiting RGB and depth data together with their computed optical flow (4 different data types), analyses the motion using a spatial focus attention, which restricts the focus on specific body parts (e.g. global, right hand, left hand). Having a total number of 12 features

3. Proposed Method

In this section, we present the mathematical formulation and the transformer-based implementation of our method. The proposed model can process an input sequence of variable length and outputs the gesture classification. An overall view of the architecture is represented in Figure 1.

3.1. Formulation

The proposed gesture recognition architecture can be defined as a function

\[
\Gamma : \mathbb{R}^{m \times w \times h \times c} \rightarrow \mathbb{R}^{n}
\]

that predicts a probability distribution over \( n \) classes from a set \( S_t \in \mathbb{R}^{m \times w \times h \times c} \) of \( m \) sequential frames \( I_t \), with size \( w \times h \) and \( c \) channels, acquired in a time range \( t \). In other words, the function \( \Gamma \) takes a sequence clip and predicts a class distribution over the considered hand gestures. The
function can be decomposed in the following three components.

The first operation corresponds to a feature extraction function $F$ applied at frame level:

$$f_t = F(S_t) \quad \text{where} \quad F : \mathbb{R}^{m \times w \times h \times c} \rightarrow \mathbb{R}^{m \times k} \quad (2)$$

Here, the extracted features $f_t$ consist of $m$ independent visual features of size $k$. Therefore, the function $F$ can be defined as the concatenation of the results of a frame-level feature extractor:

$$F(S_t) = f_t^0 \oplus f_t^1 \oplus \ldots \oplus f_t^m \quad \text{where} \quad f_t^j = G(S_t^j) \quad (3)$$

where $G : \mathbb{R}^{w \times h \times c} \rightarrow \mathbb{R}^{k}$ is a function that extracts visual features from a single frame $j$ of the sequence set $S_t$, $\oplus$ denotes the concatenation operator.

The second operation is a temporal combination and analysis of the visual features extracted through $F$. This process can be defined as

$$h_t = H(f_t) = H(F(S_t)) \quad \text{where} \quad H : \mathbb{R}^{m \times k} \rightarrow \mathbb{R}^l \quad (4)$$

where $H$ is a temporal function that processes $m$ feature maps of size $k$ and outputs an aggregated feature map of size $l$ which encodes the temporal information of $S_t$.

Finally, the last operation is a mapping between the extracted temporal features $h_t$ and the $n$ gesture classes:

$$y_t = Y(h_t) = Y(H(F(S_t))) \quad \text{where} \quad Y : \mathbb{R}^l \rightarrow \mathbb{R}^n \quad (5)$$

The resulting $y_t$, being a probability distribution over $n$ classes, is a vector of size $n$ so that $\sum_{i=1}^{n} y_{t,i} = 1$ and $y_{t,i} \in [0, 1]$.

### 3.2. Implementation

In our implementation, the function $H$ is a combination of multiple neural networks, defined as following.

The function $F$ is the concatenation of the frame-level features extracted by the function $G$, which is implemented as ResNet-18 [21], taken from the first layer up to the last convolutional and average pooling layers. The network is designed for color images, but we adapt the first layer to work with inputs having a lower number of channels $c$ as proposed in [33]. In practice, the convolutional kernels of the first layer are adapted to $1$-channel images by summing their channels. In a similar way, they are adapted to $2$-channel images by removing the third channel and rescaling the first two with a factor of $1.5$.

The function $H$, which has to temporally combine the frames of the clip $S_t$, corresponds to a slightly-modified Transformer module [43] followed by an average pooling at frame level. The model can handle sequences of any length and can be defined formally as

$$H(x) = \text{AvgPool}(\text{Encoders}(x + PE)) \quad (6)$$

where $\text{AvgPool}(\cdot)$ denotes the average pooling operation over the $m$ frames, while $\text{Encoders}(\cdot)$ corresponds to a sequence of 6 transformer encoders $E$, defined in the following.

As detailed in [43], we add positional encodings $PE$ to the input data as a way of including temporal information about the order of the frames into the model, which does not contain any recurrent module. Among the several positional encodings [19], we employ the proposal of [43].

Each transformer encoder can be defined as

$$E(x) = \text{Norm}(x + \text{FC}(\text{mhAtt}(x))) \quad (7)$$
where $\text{Norm}(\cdot)$ is a normalization layer, $\text{FC}(\cdot)$ is a sequence of two fully connected layers with 1024 units, followed by drop out (drop probability 0.1) and divided by a ReLU activation function. The multi-head attention block $\text{mhAtt}$ is a self-attention layer that can be defined as

$$\text{mhAtt}(x) = (\text{Att}_1(x) \oplus \ldots \oplus \text{Att}_8(x)) \cdot W^O$$ (8)

where

$$\text{Att}_i(x) = \text{softmax} \left( \frac{Q_i \cdot K_i}{\sqrt{d_k}} \right) \cdot V_i$$ (9)

Here, $Q_i = xW^Q_i$, $K_i = xW^K_i$, $V_i = xW^Y_i$ are independent linear projections of $x$ into a 64-d feature space, $d_k = 64$ is a scaling factor corresponding to the feature size of $K_i$, $\oplus$ is the concatenation operator and $W^O$ is a linear projection from and to a 512-d feature space.

Finally, the function $Y$ is implemented as a fully connected layer with $n$ hidden units followed by a softmax layer, resulting in a probability distribution over the $n$ classes. The predicted gesture corresponds to the class with the highest probability.

We note that the proposed approach is supposed to receive a sequence of frames containing the whole gesture or can be applied with a sliding-window approach. The temporal segmentation, i.e., the detection of the beginning and the end of each gesture, and the gesture detection, i.e., the distinction between gesture and no-gesture sequences, are out of the scope of this paper.

3.3. Data Representation

As mentioned above, we focus our investigation on the use of data produced by active depth sensors, i.e., depth data and infrared (amplitude) images. We include also RGB data since several depth devices available in the market consist of a combination of infrared and intensity sensors, like the Microsoft Kinect or Intel RealSense families.

In addition, we propose the use of surface normals, in which each pixel encodes the three components of the estimated surface normal in that point. From depth maps we obtain a representation containing an estimation of the surface normals, as introduced in [3]. Given a depth map $D$, we define $Z(x, y)$ as one of its pixel values. We compute the direction $d = \langle d_x, d_y, d_z \rangle$ of a surface normal as:

$$d = \left(-\frac{\partial Z(x, y)}{\partial x}, -\frac{\partial Z(x, y)}{\partial y}, 1 \right)$$ (10)

where $\partial Z(x, y)/\partial x, \partial Z(x, y)/\partial y$ can be considered the depth gradients in the $x$ and $y$ directions [34], or rather:

$$\frac{\partial Z(x, y)}{\partial x} = Z(x+1, y) - Z(x, y)$$

$$\frac{\partial Z(x, y)}{\partial y} = Z(x, y+1) - Z(x, y)$$ (11)

Here, $\hat{v}$ is a normalization operation [2]:

$$\hat{v} = \frac{1}{B} (d_x, d_y, 1), \ B = \sqrt{d_x^2 + d_y^2 + 1}$$ (12)

Normals computed from depth maps are not frequently used in the literature, especially in the case of the hand gesture recognition task with neural architectures. Preliminary work investigated the use of surface normals for hand pose estimation [44] or human activity recognition [49, 36]. We show in the following that this representation is complementary to the common depth images and that greatly improves the overall accuracy when used in combination with the original depth data.

In order to compare our work with literature competitors, we also compute the optical flow from consecutive RGB frames following the implementation of Farnebäck et al., [17]. It is a well-known data representation that is often used to improve the performance of the proposed system, even in the hand recognition task [33, 1], thanks to its ability to provide an estimation of the magnitude and the direction of the object (the hands in our case) motion.

3.4. Multimodal Integration

Multimodal architectures are becoming increasingly common in the literature, for a variety of different tasks. Since several input types are available from RGB-D sensors, we adopt a neural network architecture that can be easily adapted to work with a single input type or a multimodal combination of them. Specifically, the proposed architecture is able to efficiently work in a unimodal way, i.e. with a single input modality (color, depth, infrared, normals or optical flow). Moreover, two or more unimodal networks can be used at the same time through a late fusion approach [41] in which the predicted probability distributions of the single models are merged into a final classification score. Late fusion strategies are reported to present comparable or even better results with respect to the state-of-the-art in many
In our case, we adopt a late fusion strategy based on the average of the intermediate scores to predict the final classification, as follows:

$$y_t = \frac{1}{N} \sum_i Y(H(F(S_{t,i})))$$

(13)

where $N$ is the total number of tested classifiers, $S_{t,i}$ is the set of sequential frames of the $i$-th input type and $F, H$ are the functions defined in Section 3.1. Then, $Y(H(F(S_{t,i})))$ is the probability distribution of a classifier trained and tested on a specific input type.

4. Experimental Evaluation

In this section, we present the experimental setting and the results obtained on two public datasets. Then, we compare with literature methods and discuss the obtained results. Since surface normals can be considered as a different representation of depth maps, we include competitors relying on RGB-D data. In addition to the core tests with depth images and estimated surface normals, we test on color and other modalities to compare with existing literature methods.

4.1. Datasets

Being interested in the usage of depth or RGB-D sensors and in the automotive environment, in which the light invariance is a key factor, we test our approach on two datasets, NVGestures [33] and Briareo [31], collected in a car simulator.

**Nvidia Dynamic Hand Gesture.** This dataset [33], also called NVGestures, is the largest dynamic hand gesture dataset in an automotive setting, in terms of number of gestures, subjects and sequences. Video sequences are acquired with two sensors: the SoftKinetic DS325, an active RGB-D sensor, and the DUO 3D, an infrared stereo camera. These acquisition devices lead to 3 modalities (RGB, depth, IR) and 5 streams (color, depth, color mapped on depth, IR left, IR right), available in the dataset. NVGestures is acquired in an indoor car simulator, the depth camera is placed next to the infotainment system, while the stereo camera is placed on top of the acquisition area. Authors did not release the infrared amplitude recorded by the depth sensor, but they provided infrared data from the dedicated DUO 3D camera, placed in a different position. The dataset contains 25 different gestures performed by 20 subjects with the right hand. Each gesture is repeated three times and acquired in 5-second video samples. Gestures range from swipes to rotations and from showing $n$ fingers to showing the “OK” sign. For further details about this dataset, please refer to the original paper [33].

In our experiments, we employ the color (RGB), depth, and infrared (left IR) modalities. In addition, we compute an estimation of the surface normals from the depth data (see Section 3.3) and we report visual samples of these data in Figure 2. In order to compare with literature work, we compute the optical flow on color frames through [17], as done in previous work [33].

**Briareo.** This is a recently-released automotive dataset [31] for the dynamic hand gesture recognition task. Video sequences are acquired using three synchronized devices: an active depth sensor (Pico Flexx), an infrared stereo sensor (Leap Motion) and a standard RGB camera. Therefore, several image types are available: depth and infrared amplitude, left and right IR, color. In addition, the SDK of the Leap Motion device has been used to estimate and record the hand joint positions. Recording devices are placed in the central tunnel console of a car simulator between the driver and the passenger seat, looking upwards. In this case, authors released the infrared amplitude recorded by the depth sensor, along with the infrared data acquired by the Leap Motion sensor. The dataset contains 12 different gestures performed by 40 different subjects (33 males and 7 females) with the right hand. As in NVGestures, each gesture is repeated three times and captures the entire gesture motion. Gestures are designed for the interaction between the driver and the car infotainment system. Some examples are the swipes in the four directions and the “thumb up” and “phone” signs. For further details about this dataset, please refer to the original paper [31].

In our experiments, we employ most of the available modalities, i.e. color (RGB), depth, and infrared amplitude. In addition, we estimate the surface normals from the depth data, as explained in Section 3.3 and depicted in Figure 3.

4.2. Model Training

We train and test the model with fixed-length clips of 40 frames extracted from the dataset sequences around the
center of the gesture. We empirically set this input size, but the proposed model can potentially analyze sequences of any length thanks to its flexible design. For the NVGestures dataset, we extract the 80 central frames around the gesture and sample them to obtain 40 equidistant frames. For the Briareo dataset, which has a lower frame rate, we select the 40 frames containing the gesture movement.

Each input data is normalized individually to obtain zero mean and unit variance input, with the exception of the surface normals that are normalized to have unit-magnitude and are contained in the range $[-1, 1]$. Then, frames are cropped to $224 \times 224$ pixels as required by the chosen frame-level feature extractor (i.e. ResNet-18). We apply random rescale (with rescale factor in the range $[0.8, 1.2]$), random crop and random rotation between $-15$ and $15$ degrees as data augmentation, in order to avoid overfitting.

The ResNet-18 architecture is initialized with weights pre-trained on ImageNet [11] while the remaining of the architecture is trained from scratch. The architecture is then trained end-to-end using the Adam optimizer [24] to minimize the categorical cross entropy loss. We use a mini-batch size of 8 video samples, learning rate $1e^{-4}$, weight decay $1e^{-4}$ and random dropout. We apply the early stopping based on the accuracy on the validation set, following the official dataset splits.

A different model is trained for each modality and multiple modalities are combined at prediction level with the late fusion approach presented in Section 3.4. Empirically, we find that other types of fusion, e.g. mid and early fusion, results in overfitting on the training set, in line with what found in [33].

### 4.3. Results using NVGestures dataset

We analyze here the performance on the NVGestures dataset.

Table 1 compares our method to the literature in the unimodal case, i.e. when a single input is fed into the model. Focusing on depth data, the proposed approach achieves state-of-the-art results when depth maps are the only used input. A similar high accuracy is also achieved using surface normals as input, revealing that normals are a discriminative representation for the hand gesture recognition task, even though no competitors are currently available. Also the infrared modality overcomes the competitor, even if the final accuracy is lower. On the other remaining modalities, i.e. color and optical flow, our method achieves comparable accuracy to the I3D method [8, 1]. However, we note this method is pre-trained on ImageNet [11] (as our feature extractor) and on Kinetics [23], which is a large dataset of action recognition in videos. We hypothesize that the slight gap between this and our method can be due to this pre-training step, which was not available for the other types of the exploited data.

<table>
<thead>
<tr>
<th>Method</th>
<th>Modality</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Spat. st. CNN [40]</td>
<td>54.6%</td>
</tr>
<tr>
<td></td>
<td>iDT-HOG [45]</td>
<td>59.1%</td>
</tr>
<tr>
<td>color</td>
<td>Res3ATN [12]</td>
<td>62.7%</td>
</tr>
<tr>
<td></td>
<td>C3D [42]</td>
<td>69.3%</td>
</tr>
<tr>
<td></td>
<td>R3D-CNN [33]</td>
<td>74.1%</td>
</tr>
<tr>
<td></td>
<td>Ours</td>
<td>76.5%</td>
</tr>
<tr>
<td></td>
<td>I3D [8]†</td>
<td>78.4%</td>
</tr>
<tr>
<td>depth</td>
<td>SNV [47]</td>
<td>70.7%</td>
</tr>
<tr>
<td></td>
<td>C3D [42]</td>
<td>78.8%</td>
</tr>
<tr>
<td></td>
<td>R3D-CNN [33]</td>
<td>80.3%</td>
</tr>
<tr>
<td></td>
<td>I3D [8]†</td>
<td>82.3%</td>
</tr>
<tr>
<td></td>
<td>Ours</td>
<td>83.0%</td>
</tr>
<tr>
<td>infrared</td>
<td>R3D-CNN [33]</td>
<td>63.5%</td>
</tr>
<tr>
<td></td>
<td>Ours</td>
<td>64.7%</td>
</tr>
<tr>
<td>flow</td>
<td>iDT-HOF [45]</td>
<td>61.8%</td>
</tr>
<tr>
<td></td>
<td>Temp. st. CNN [40]</td>
<td>68.0%</td>
</tr>
<tr>
<td></td>
<td>Ours</td>
<td>72.0%</td>
</tr>
<tr>
<td></td>
<td>iDT-MBH [45]</td>
<td>76.8%</td>
</tr>
<tr>
<td></td>
<td>R3D-CNN [33]</td>
<td>77.8%</td>
</tr>
<tr>
<td></td>
<td>I3D [8]†</td>
<td>83.4%</td>
</tr>
<tr>
<td>normals</td>
<td>Ours</td>
<td>82.4%</td>
</tr>
<tr>
<td>color</td>
<td>Human [33]</td>
<td>88.4%</td>
</tr>
</tbody>
</table>

Table 1. Unimodal results on NVGestures [33]. Previous results are taken from the respective papers and from [33, 1]. † indicates models pre-trained on Kinetics [23], in addition to ImageNet [11].

Moving from the unimodal to the multimodal case, we show in Table 2 a thorough analysis of the possible multimodal combinations, following the late-fusion approach reported in Section 3.4. The results are grouped by number of employed modalities and ordered by accuracy. It can be seen that, in general, the proposed approach benefits from the multimodal integration. Moreover, the best performing methods in each group are those using a combination of depth and surface normals as input data, confirming that the partial 3D data obtained by the depth sensors contains discriminative information for the gesture recognition task.

We highlight that the combination of depth images and surface normals leads to a remarkable accuracy of 87.3%. This result confirms that these two modalities are complementary and their combination greatly improves the overall accuracy compared to the usage of a single modality (which scores 83.0 for the depth and 82.4 for the surface normals). Combining additional modalities (color and infrared) the accuracy is slightly incremented, reaching 87.6%.

We also compare our method in the multimodal setting with state-of-the-art approaches reported in Table 3. Among other methods that exploit several data types, our approach
Table 2. Multimodal results on NVGestures [33] using several combinations of modalities. # refers to the number of used modalities.

<table>
<thead>
<tr>
<th>#</th>
<th>Modality</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>infrared (ir)</td>
<td>64.7%</td>
</tr>
<tr>
<td></td>
<td>color</td>
<td>76.5%</td>
</tr>
<tr>
<td></td>
<td>normals</td>
<td>82.4%</td>
</tr>
<tr>
<td></td>
<td>depth</td>
<td>83.0%</td>
</tr>
<tr>
<td>2</td>
<td>color + ir</td>
<td>79.0%</td>
</tr>
<tr>
<td></td>
<td>depth + ir</td>
<td>81.7%</td>
</tr>
<tr>
<td></td>
<td>normals + ir</td>
<td>82.8%</td>
</tr>
<tr>
<td></td>
<td>color + depth</td>
<td>84.6%</td>
</tr>
<tr>
<td></td>
<td>color + normals</td>
<td>84.6%</td>
</tr>
<tr>
<td></td>
<td>depth + normals</td>
<td>87.3%</td>
</tr>
<tr>
<td>3</td>
<td>color + ir + depth</td>
<td>85.3%</td>
</tr>
<tr>
<td></td>
<td>color + ir + normals</td>
<td>85.3%</td>
</tr>
<tr>
<td></td>
<td>color + depth + normals</td>
<td>86.1%</td>
</tr>
<tr>
<td></td>
<td>depth + normals + ir</td>
<td>87.1%</td>
</tr>
<tr>
<td>4</td>
<td>color + depth + normals + ir</td>
<td>87.6%</td>
</tr>
</tbody>
</table>

Table 3. Multimodal results on NVGestures [33], comparison with competitors. Previous results are taken from the respective papers and from [33, 1]. † indicates models pre-trained on Kinetics [23], in addition to ImageNet [11], while * shows models pre-trained on the Jester gesture dataset [32].

<table>
<thead>
<tr>
<th>Method</th>
<th>Modality</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Two-st. CNNs [40]</td>
<td>color + flow</td>
<td>65.6%</td>
</tr>
<tr>
<td>iDT [45]</td>
<td>color + flow</td>
<td>73.4%</td>
</tr>
<tr>
<td>R3D-CNN [33]</td>
<td>color + flow</td>
<td>79.3%</td>
</tr>
<tr>
<td>R3D-CNN [33]</td>
<td>color + depth + flow</td>
<td>81.5%</td>
</tr>
<tr>
<td>R3D-CNN [33]</td>
<td>color + depth + ir</td>
<td>82.0%</td>
</tr>
<tr>
<td>R3D-CNN [33]</td>
<td>depth + flow</td>
<td>82.4%</td>
</tr>
<tr>
<td>R3D-CNN [33]</td>
<td>all</td>
<td>83.8%</td>
</tr>
<tr>
<td>8-MFFs-3f1c [26]*</td>
<td>color + flow</td>
<td>84.7%</td>
</tr>
<tr>
<td>I3D [8]†</td>
<td>color + depth</td>
<td>83.8%</td>
</tr>
<tr>
<td>I3D [8]†</td>
<td>color + flow</td>
<td>84.4%</td>
</tr>
<tr>
<td>I3D [8]†</td>
<td>color + depth + flow</td>
<td>85.7%</td>
</tr>
<tr>
<td>MTUT_RGB-D [1]†</td>
<td>color + depth</td>
<td>85.5%</td>
</tr>
<tr>
<td>MTUT_RGB-D+flow [1]†</td>
<td>color + depth</td>
<td>86.1%</td>
</tr>
<tr>
<td>MTUT_RGB-D+flow [1]†</td>
<td>color + depth + flow</td>
<td>86.9%</td>
</tr>
<tr>
<td>Ours</td>
<td>depth + normals</td>
<td>87.3%</td>
</tr>
<tr>
<td>Ours</td>
<td>color + depth + normals + ir</td>
<td>87.6%</td>
</tr>
<tr>
<td>Human [33]</td>
<td>color</td>
<td>88.4%</td>
</tr>
</tbody>
</table>

4.4. Results using Briareo dataset

Table 4 presents the results of the unimodal and the multimodal setting for the Briareo dataset. The results are grouped by number of employed modalities and ordered by accuracy.

Considering the unimodal case, the surface normals obtains the highest accuracy, reaching 95.8%, outperforming the results using other modalities. This confirms that surface normals estimated from depth are an informative and discriminative representation for the hand gesture recognition task. Also the infrared source achieves a high accuracy, probably
Table 4. Unimodal and multimodal results obtained on Briareo. # refers to the number of used modalities.

<table>
<thead>
<tr>
<th>#</th>
<th>Modality</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>color</td>
<td>90.6%</td>
</tr>
<tr>
<td></td>
<td>depth</td>
<td>92.4%</td>
</tr>
<tr>
<td></td>
<td>ir</td>
<td>95.1%</td>
</tr>
<tr>
<td></td>
<td>normals</td>
<td>95.8%</td>
</tr>
<tr>
<td>2</td>
<td>color + depth</td>
<td>94.1%</td>
</tr>
<tr>
<td></td>
<td>depth + ir</td>
<td>95.1%</td>
</tr>
<tr>
<td></td>
<td>color + ir</td>
<td>95.5%</td>
</tr>
<tr>
<td></td>
<td>depth + normals</td>
<td>96.2%</td>
</tr>
<tr>
<td></td>
<td>color + normals</td>
<td>96.5%</td>
</tr>
<tr>
<td></td>
<td>ir + normals</td>
<td>97.2%</td>
</tr>
<tr>
<td>3</td>
<td>color + depth + ir</td>
<td>95.1%</td>
</tr>
<tr>
<td></td>
<td>color + depth + normals</td>
<td>95.8%</td>
</tr>
<tr>
<td></td>
<td>color + ir + normals</td>
<td>96.9%</td>
</tr>
<tr>
<td>4</td>
<td>color + depth + ir + normals</td>
<td>97.2%</td>
</tr>
</tbody>
</table>

due to the position of the infrared sensor, close to the hand. The combination of multiple modalities, with the late fusion approach presented in Section 3.4, slightly improves the overall results. The fusion of infrared and normals results in an overall accuracy of 97.2% which is the highest result. While the combination of surface normals with infrared and depth increases the combined accuracy, the usage of color data does not provide significant gains.

In Table 5 we compare our method in the multimodal setting with state-of-the-art approaches. The proposed approach obtains state-of-the-art accuracy 97.2% using only infrared data and surface normals, which derive from a single active depth sensor. Even with the usage of a single modality, e.g. surface normals, our method outperforms the literature competitors by a clear margin. Indeed, it performs better than methods based on recurrent networks (LSTMs) and 3D joint features (computed by the Leap Motion SDK), which require additional computation. Also in this case, the whole system requires a single active depth device and can run in real time, as shown in the next section.

### 4.5. Performance Analysis

We assess the computational requirements of our and other architectures in terms of number of parameters, inference time on a single GPU, and required VRAM on the graphics card. We test them on a workstation with an Intel Core i7-7700K and a Nvidia GeForce GTX 1080 Ti. As shown in Table 6, our method has fewer parameters, faster inference speed and comparable memory usage when used with a single modality. When applied on multiple modalities, running in parallel on the same hardware, the proposed approach still maintains real time speed and acceptable memory usage, both in case of 2 modalities and in case of 4 modalities.

<table>
<thead>
<tr>
<th>Method</th>
<th>Modality</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>C3D-HG [31]</td>
<td>color</td>
<td>72.2%</td>
</tr>
<tr>
<td>C3D-HG [31]</td>
<td>depth</td>
<td>76.0%</td>
</tr>
<tr>
<td>C3D-HG [31]</td>
<td>ir</td>
<td>87.5%</td>
</tr>
<tr>
<td>LSTM-HG [31]</td>
<td>3D joint features</td>
<td>94.4%</td>
</tr>
<tr>
<td>Ours</td>
<td>normals</td>
<td>95.8%</td>
</tr>
<tr>
<td>Ours</td>
<td>depth + normals</td>
<td>96.2%</td>
</tr>
<tr>
<td>Ours</td>
<td>ir + normals</td>
<td>97.2%</td>
</tr>
</tbody>
</table>

Table 5. Comparison with the state-of-the-art methods tested on Briareo.

<table>
<thead>
<tr>
<th>Model</th>
<th>Parameters (M)</th>
<th>Inference (ms)</th>
<th>VRAM (GB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>R3D-CNN [33]</td>
<td>38.0</td>
<td>30</td>
<td>1.3</td>
</tr>
<tr>
<td>C3D-HG [31]</td>
<td>26.7</td>
<td>55</td>
<td>1.0</td>
</tr>
<tr>
<td>Ours (1 modality)</td>
<td>24.3</td>
<td>26.7</td>
<td>1.8</td>
</tr>
<tr>
<td>Ours (2 modalities)</td>
<td>48.6</td>
<td>61.7</td>
<td>3.0</td>
</tr>
<tr>
<td>Ours (4 modalities)</td>
<td>97.2</td>
<td>108.3</td>
<td>5.3</td>
</tr>
</tbody>
</table>

Table 6. Performance analysis of the proposed method. Specifically, we report the number of parameters, the inference time and the amount of video RAM (VRAM) needed to run the system.

5. Conclusions

In this paper, we propose a transformer-based architecture for the dynamic hand gesture recognition task. Through an extensive evaluation we show how the frame-level feature extraction and the temporal aggregation computed by the transformer, starting from depth and surface normals combined through a late fusion approach, achieves state-of-the-art results. Moreover, we investigate the use of other data types usually provided by RGB-D sensors, such as color and infrared images. Experimental results obtained on two automotive datasets, namely NVidia Dynamic Hand Gesture and Briareo, confirm the feasibility of the proposed method for the automotive setting, in which the light invariance is an enabling element. Even though the temporal flow is explicitly encoded into the transformer-based architecture, there are several “symmetric” gestures that are occasionally confused. In fact, the main challenges of the problem are still related to the temporal progression of the gesture, which will be addressed in future work. The performance analysis shows that the framework is able to run with real time performance and it requires a limited amount of video memory, making it suitable for an online infotainment system.
References


