CGAPoseNet+GCAN: A Geometric Clifford Algebra Network for Geometry-aware Camera Pose Regression

Alberto Pepe, Joan Lasenby
Signal Processing and Communications Lab
University of Cambridge
{ap2219, jl221}@cam.ac.uk

Sven Buchholz
Department of Computer Science and Media
Technical University Brandenburg
sven.buchholz@th-brandenburg.de

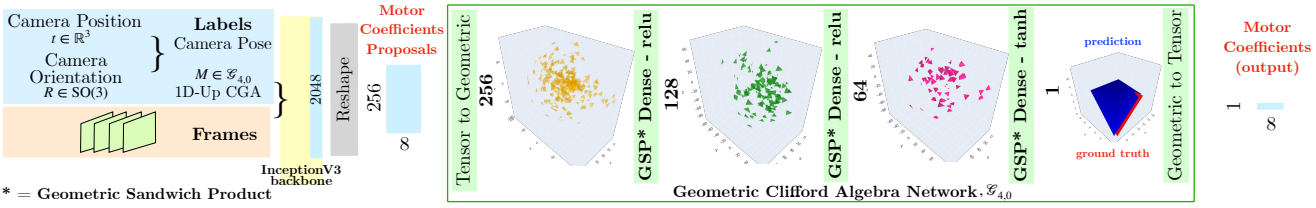


Figure 1. The CGAPoseNet+GCAN architecture. The output of the InceptionV3 network is reshaped to obtain a set of motor coefficients proposals. Motors are objects in the 1D-Up Conformal Geometric Algebra (CGA) $\mathcal{G}_{4,0}$ with scalar, bivector and quadrivector parts, giving a total of 8 real coefficients. These coefficients are used to build motors $\in \mathcal{G}_{4,0}$ in input to the Geometric Clifford Algebra Network (GCAN). A motor represents a rotation and a translation, and it is hence a suitable representation for camera poses. The GCAN works in $\mathcal{G}_{4,0}$ space and has weights, biases and outputs that are also motors, and hence interpretable as poses. The GCAN narrows down the proposals to a single motor through a geometric understanding of the scene.

Abstract

We introduce CGAPoseNet+GCAN, which enhances CGAPoseNet, an architecture for camera pose regression, with a Geometric Clifford Algebra Network (GCAN). With the addition of the GCAN we obtain a geometry-aware pipeline for camera pose regression from RGB images only. CGAPoseNet employs Clifford Geometric Algebra to unify quaternions and translation vectors into a single mathematical object, the motor, which can be used to uniquely describe camera poses. CGAPoseNet can obtain comparable results to other approaches without the need of expensive tuning of the loss function or additional information about the scene, such as 3D point clouds, which might not always be available. CGAPoseNet, however, like several approaches in the literature, only learns to predict motor coefficients, and it is unaware of the mathematical space in which predictions sit in and of their geometrical meaning. By leveraging recent advances in Geometric Deep Learning, we modify CGAPoseNet with a GCAN: proposals of possible motor coefficients associated with a camera frame are obtained from the InceptionV3 backbone, and the GCAN downsamples them to a single motor through a sequence of layers that work in $\mathcal{G}_{4,0}$. The network is hence geometry-aware, has multivector-valued inputs, weights and biases and preserves the grade of the objects that it receives in input. CGAPoseNet+GCAN has almost 4 million fewer trainable parameters, it reduces the average rotation error by 41% and the average translation error by 8.8% compared to CGAPoseNet. Similarly, it reduces rotation and translation errors by 32.6% and 19.9%, respectively, compared to the best performing PoseNet strategy. CGAPoseNet+GCAN reaches the state-of-the-art results on 13 commonly employed datasets. To the best of our knowledge, it is the first experiment in GCANs applied to the problem of camera pose regression.

1. Introduction

Camera pose regression is the process of estimating the 3D position and orientation (i.e. the pose) of a camera relative to a given object or scene. It has found application in augmented reality [45, 56, 71], object tracking [34, 55, 75], localization and mapping [5, 25, 73] and three-dimensional (3D) reconstruction [1, 40, 83].

Before deep learning (DL), camera pose regression was performed through traditional computer vision algorithms. These include: (i) feature-based algorithms, such as SIFT [54] or SURF [2], which extract the camera pose by matching features (edges, corners, *ad hoc* descriptors, etc.) across

multiple views of the scene and triangulating them; (ii) iterative methods, such as Perspective-n-Point (PnP) [27] and Bundle Adjustment (BA) [74], that minimize an objective function such as the reprojection error between 2D points on the image and 3D points in space; (iii) structure from motion [28, 31], that jointly reconstructs the 3D geometry from multiple views and estimates the camera pose. These methods are generally very accurate, but they require special handling of outliers (e.g. through the RANSAC algorithm [27]), precisely crafted features and they generally fail under large viewpoint changes or in presence of occlusions. One of the earliest examples of DL approaches to camera pose regression is found in [43], in which information about the scene is extracted directly from the RGB images with a convolutional neural network (CNN), which does not require hand-crafted feature descriptors. Despite the success of CNN approaches in the literature [6, 7, 41, 42, 57, 80] we believe that most pose regression problems via DL suffer from two major drawbacks: (i) they require a separate handling of rotation and translation components as they sit in two different mathematical spaces (ii) they are geometry-agnostic and ignore the structure of the scene being recorded, unlike more traditional computer vision approaches.

In this paper we expand CGAPoseNet, presented in [60], by adding a Geometric Clifford Algebra Network (GCAN) [66] to it. The GCAN sits in the same mathematical space of the predictions (see Figure 1). The use of motors in CGAPoseNet solves the separate treatment of rotation and translation component, but CGAPoseNet, like most regression problems, has the limitation of predicting poses without knowledge about the geometry of the scene. With CGAPoseNet+GCAN we let the backbone predict a set of *proposals* of motor coefficients (rather than a single motor as CGAPoseNet does) which are then transformed into *actual* motors and fed into the GCAN, which operates on them. This enables a geometry-aware approach in which inputs and outputs of the GCAN layers are *also* camera poses. This allows for better understanding of the geometry of the scene, better generalizability on prevously unseen data and better interpretability of the intermediate layers' outputs.

Our CGAPoseNet+GCAN architecture significantly reduces the translation and rotation error with respect to both CGAPoseNet, from which we borrow the pose representation and the loss function, and PoseNet with geometric reprojection error loss, which is the best performing PoseNet strategy. Moreover, CGAPoseNet+GCAN adds only a few trainable parameter to the base PoseNet approach, but with a simpler MSE loss function to be minimized and no additional information about the 3D scene required (e.g., no 3D point cloud information is necessary), and it has about 4 million fewer parameters than CGAPoseNet.

2. Related Work

2.1. Geometric Algebra

In this section we will provide some basics of Geometric Algebra (GA) needed to follow the approach presented. For a more complete introduction, we refer the reader to [23,35,36,50]. GA is the term introduced by David Hestenes [35] as a form of Clifford Algebra which offers a unifying language for applied mathematics. GA has found application in general relativity [35], quantum mechanics [70], computer vision [78], computer graphics [29], computational chemistry [20, 51] and bioinformatics [61, 63].

The Geometric Algebra we work with is a real Clifford algebra. Given an n-dimensional real vector space $\mathbb{R}^n$, we define a GA $\mathcal{G}_{p,q,r}$, with $n = p + q + r$, with p basis vectors that square to 1, q basis vectors that square to -1 and r basis vectors that square to 0. $\mathcal{G}_{3,0,0}$, for example, is the 3D Euclidean GA, and it is spanned by $\{1, e_1, e_2, e_3, e_{12}, e_{13}, e_{23}, e_{123}\}$. The *grade* of an object in GA refers to the dimensionality of the subspace it defines, so $\{1\}$, a scalar or a 0-*blade*, has grade 0, $\{e_1, e_2, e_3\}$, vectors or 1-*blades*, have grade 1, $\{e_{12}, e_{13}, e_{23}\}$, *bivectors* or 2-*blades*, have grade 2 and $\{e_{123}\}$, a *trivector* or 3-*blade*, has grade 3. Multiplication of vectors in GA yields objects called *multivectors*, which are higher-dimensional objects.

GA is built out of two fundamental operations, addition and the geometric product, which, given two GA vectors $\mathbf{u}, \mathbf{v}$, is defined as

$$\mathbf{uv} = \mathbf{u} \cdot \mathbf{v} + \mathbf{u} \wedge \mathbf{v} \tag{1}$$

in which $\cdot$ indicates the inner product and $\wedge$ indicates the Grassman outer product. $\mathbf{uv}$ is a multivector, as it is a linear combination of objects with different grades, i.e. a scalar ($\mathbf{u} \cdot \mathbf{v}$) and a bivector ($\mathbf{u} \wedge \mathbf{v}$). The geometric product is associative, distributive and closed under multiplication, for which $\mathbf{uv} \in \mathcal{G}_{p,q,r}$.

Rotors. Given a geometric product of vectors $R = \mathbf{u}_1\mathbf{u}_2...\mathbf{u}_k$ in an nD space where $k \leq n$, we define the reversion operator $\tilde{R} = \mathbf{u}_k\mathbf{u}_{k-1}...\mathbf{u}_1$. By scaling R so that $R\tilde{R} = 1$, then we define

$$\mathbf{v}' = R\mathbf{v}\tilde{R} \tag{2}$$

to be a "sandwich" product, i.e. the geometric product of a GA object in between a rotor and its reverse. Equation 2 represents a rotoreflection or, if k is even, a rotation. In the latter case, we call R a *rotor*. Rotors are isomorphic to quaternions in 3D, and they can be regarded as generalized quaternions for any nD space.

2.2. Geometric Algebra Neural Networks

The concept of geometric deep learning was introduced in [11]. The idea behind it is to preserve the inherent ge-

ometry and structure of data and extract information from it through *ad hoc* models and algorithms. There are many examples of the disruptive power of geometric deep learning in the literature [17, 26, 58, 81], but here we focus on GA-based strategies.

GA is an intuitive framework to represent, manipulate and transform geometric objects such as points, lines, planes and spheres. Hence, several attempts have been made to build geometric DL models that operate in GA in order to capture the geometric nature of data and achieve equivariance, i.e., we obtain the same output if we apply a geometric transformation to the input function to the neural network or we transform the output. GA for neural computation was first introduced in [59]. Multivector-valued neurons, as opposed to real-valued neurons, were later presented for a radial basis function network [21], for the multilayer perceptron (MLP) [12, 14] and in neural networks (NNs) [4, 15, 16].

As of today, GA NNs have found application to several problems, including signal processing [13], robotics [3], PDE modeling [8], fluid dynamics [66] and particle physics [65]. Moreover, several new GA-based architectures have been introduced, including multivector-valued CNNs [52, 77], recurrent neural networks [46, 84] and transformer networks [10, 53].

The architecture we employed in this paper, the Geometric Clifford Algebra Network (GCAN), is directly derived from [66]. GCANs parametrize linear combinations of learnable group actions, meaning that GCANs are trainable and adjustable *geometric templates*, which excel at modeling rigid body transformations. The advantage of GCANs is that they ensure covariance at the layer level and that they preserve the grade of input objects. This is enforced by substituting standard layers with sandwich product layers, that in GA represent rigid geometric transformations.

2.3. Pose parametrization

Camera pose regression means predicting, in a supervised fashion, the camera pose $\mathbf{p} \in \mathrm{SE}(3)$, with $\mathrm{SE}(3) \triangleq \{(\mathbf{R}, \mathbf{t}) : \mathbf{R} \in \mathrm{SO}(3), \mathbf{t} \in \mathbb{R}^3\}$ for a given frame of a video capture of a scene. The translation component is generally represented as a 3D vector in $\mathbb{R}^3$. The rotation component, on the other hand, can be parametrized in multiple ways including rotation matrices, quaternions, Euler angles, axis-angles representations, rotors, bivectors and more.

The impact of the rotation representation in machine learning has been widely studied [9, 18, 64, 79]. The gimbal lock of Euler angles or the double coverage of quaternions, for example, negatively impact the regression quality. The discontinuity in the mapping from the rotation matrix $\mathbf{R} \in \mathrm{SO}(3)$ onto a given representation space has also been highlighted as a limiting factor [62, 67, 82].

-	PoseNet	CGAPoseNet	CGAPoseNet +GCAN
Parameters	21,782,695	25,918,224	22,132,520

Table 1. Number of trainable parameters for the three approaches.

Hence, in camera pose regression problems, two things have to be taken into account: (i) the choice of a rotation representation suitable for the learning algorithm and (ii) the weighting of the translation and rotation components. In PoseNet [43], rotations are expressed as quaternions and they are weighted with the translation through a scalar coefficient β in the loss function:

$$\mathcal{L}_\beta = \mathcal{L}_\mathbf{t} + \beta \mathcal{L}_\mathbf{q} \tag{3}$$

in which $\mathcal{L}_\mathbf{t}$ and $\mathcal{L}_\mathbf{q}$ are the translation and rotation loss, resepctively. The choice of β depends on the dataset and the kind of architecture employed. The value of β requires expensive tuning through a grid search and cannot be intuitively picked based on geometric information of the dataset. Similar weighting strategies are found in [24, 76]

Two more advanced loss functions, that leverage geometry information, have been presented in [42], namely (i) a probabilistic DL approach, using homoscedastic uncertainty as a weigthing factor, and (ii) a weighting-free approach via geometric reprojection error. Both approaches significantly improve the results compared to baseline PoseNet [43] as they include geometry information of the scene, but approach (i) still represents a cumbersome tuning of loss functions over objects sitting in different spaces and approach (ii) requires additional information about the 3D points of the scene, which is not always available.

A unifying approach to modelling rotations and translations was proposed with CGAPoseNet in [60]. Leveraging the PoseNet pipeline, CGAPoseNet unifies rotations and translations with motors in a curved space, which then only requires an MSE loss. CGAPoseNet, however, ignores the geometry of the scene as it only predicts motor coefficients.

3. Methodology

3.1. 1D-Up CGA

We represent poses in $\mathcal{G}_{4,0,0}$ (which we will refer to as $\mathcal{G}_{4,0}$). The $\mathcal{G}_{4,0}$ algebra is called the 1D-Up CGA because it is a Conformal Geometric Algebra (CGA) with only 1 extra dimension, i.e. we are modelling a 3D space with a 4D algebra [47–49]. CGA, on the other hand, extends a GA $\mathcal{G}_{p,q}$ to $\mathcal{G}_{p+1,q+1}$, hence it requires 2 extra dimensions. $\mathcal{G}_{4,0,0}$ has four basis vectors $\{e_1, e_2, e_3, e_4\}$, for which $e_i^2 = +1$ $\forall\, i \in \{1, 2, 3, 4\}$. The $\mathcal{G}_{4,0}$ space has constant curvature λ and it represents a spherical geometry. While it may seem counter-intuitive at first, modelling the real world in spherical space allows for: (i) a Euclidean signature space, which

is likely the main reason behind the speedy convergence of the loss during training (ii) a representation for the pose with few parameters as only 1 extra dimension is needed.

A point $x \in \mathcal{G}_{3,0}$, is mapped to $X \in \mathcal{G}_{4,0}$ through the function $f : x \to X$

$$X = f(x) = \left(\frac{2\lambda}{\lambda^2 + x^2}\right) x + \left(\frac{\lambda^2 - x^2}{\lambda^2 + x^2}\right) e_4. \quad (4)$$

It can be shown that translating and rotating in $\mathcal{G}_{4,0}$ can *both* be done through rotors. Given a translation vector $\mathbf{t} \in \mathcal{G}_{3,0}$, its corresponding rotor in 4D spherical geometry is given by:

$$T = g(\mathbf{t}) = \frac{\lambda + \mathbf{t}e_4}{\sqrt{\lambda^2 + \mathbf{t}^2}} \quad (5)$$

A rotor R in 3D Euclidean geometry is still R in 4D spherical geometry. The rigid body motion, i.e. translation *and* rotation, of an object X into X' in the 1D-Up CGA can hence be expressed as the combination of two sandwich products :

$$X' = TRX\tilde{R}\tilde{T} = MX\tilde{M} \quad (6)$$

The geometric product $M = TR$ yields a *motor*, which represents a rotation and a translation. Note how rotations and translations are now expressed in the same units. Motors are objects (multivectors) in $\mathcal{G}_{4,0}$ with only even blades, presenting 1 scalar, 6 bivector and 1 quadrivector components:

$$\begin{aligned} M = & \underbrace{x_0 1}_{\text{scalar}} \\ & + \underbrace{x_{12}e_{12} + x_{13}e_{13} + x_{14}e_{14} + x_{23}e_{23} + x_{24}e_{24} + x_{34}e_{34}}_{\text{bivector}} \\ & + \underbrace{x_{1234}e_{1234}}_{\text{quadrivector}} \end{aligned} \quad (7)$$

Since motors combine translations and rotations, they can be employed as a pose representation with 8 parameters (i.e. the 8 coefficients). An object in 1D-Up CGA X can be projected back onto 3D space via:

$$\begin{aligned} x = f^{-1}(X) = \frac{\lambda}{1 + X \cdot e_4}[(X \cdot e_1)e_1 + (X \cdot e_2)e_2 \\ + (X \cdot e_3)e_3] \end{aligned} \quad (8)$$

3.2. Architecture

The key element in our approach is the GCAN added at the output of the backbone. We call our architecture CGAPoseNet+GCAN because, like in CGAPoseNet, we also represent poses with motors in 1D-Up CGA, which unify translation and rotation with a single object that sits in a space with Euclidean signature, and we keep the mean squared error (MSE) function as a loss to guide the training.

CGAPoseNet, however, is not really working in 1D-Up CGA space as it only learns to predict poses expressed as motors *coefficients*. It does so based on patterns in the data, without understanding the poses' geometrical meaning or learning how to perform geometrical transformations on them. We believe that this is a key limitation of the CGAPoseNet approach, that explains why it does not significantly surpass PoseNet paired with geometric reprojection error, that includes information about 3D points of the scene in its loss function and therefore is a proper geometry-aware approach.

We modify CGAPoseNet by reshaping the penultimate layer of the backbone from 2048 into 256×8. We refer to this output as motor *proposals*, since the backbone now predicts 256 sets of 8 motor coefficients rather than a single set of motor coefficients as in CGAPoseNet. Proposals are then employed to build motors M_i and fed in as input to the GCAN. The GCAN explicitly works in $\mathcal{G}_{4,0}$ and it consists of 3 sandwich product dense layers, whose outputs obey

$$h(M) = \sum_{i=1}^{c} W_i M_i \tilde{W}_i + B_i \quad (9)$$

where c is the number of channels, $M = \{M_i\}_{i=1}^{c}$ is the set of motors per channel, W_i are the weights and B_i the biases. Note that we employ the uppercase notation since $W_i, M_i, B_i \in \mathcal{G}_{4,0}$ and all of them only contain even blades. This means that (i) each neuron in the layer encodes a geometric transformation of its input, preserving the grade of the objects as described in Section 2.1, and hence (ii) each output of the GCAN layers is $also$ a (unnormalized) motor in 1D-Up CGA. The GCAN layers have 128, 64 and 1 neurons, respectively: the 256 *proposals* are progressively downsampled until the optimal pose is found (see Figure 1). A pipeline with 128-64-32-1 neurons has also been tested, without significant difference.

We also slightly adapted the backbone in order to reduce the number of trainable parameters (see Table 1). CGAPoseNets adds two dense layers to the backbone, with 2048 and 8 neurons, respectively, without removing the last classification layer of InceptionV3, that has 1000 neurons (see Figure 2). This bottleneck significantly increases the number of parameters. In CGAPoseNet+GCAN, we remove the classification layer with 1000 neurons and instead reshape the 2048 outputs that precede the classification layer.

4. Experiments

4.1. Datasets

We followed [41–43, 60] and tested our approach on datasets of both indoor and outdoor scenes, for a total of

Table 2. Median translation and rotation errors over the test set for the 7 approaches.

Scene	PoseNet [43]	Bayesian PoseNet [41]	PoseNet LSTM [76]	PoseNet σ^2 Weights [42]	PoseNet Geom. Repr. [42]	CGA-PoseNet	CGA-PoseNet+GCAN
Great Court	-	-	-	7.00m, 3.65°	6.83m, 3.47°	3.77m, 4.27°	3.88m, **3.21°**
King's	1.92m, 5.40°	1.74m, 4.06°	0.99m, 3.65°	0.99m, 1.06°	**0.88m**, **1.04°**	1.36m, 1.85°	1.00m, 1.16°
Old Hospital	2.31m, 5.38°	2.57m, 5.14°	**1.51m**, 4.29°	2.17m, 2.94°	3.20m, 3.29°	2.52m, 2.90°	1.79m, **2.28°**
Shop	1.46m, 8.08°	1.25m, 7.54°	1.18m, 7.44°	1.05m, 3.97°	0.88m, 3.78°	**0.74m**, 5.84°	1.19m, **3.43°**
St. Mary's	2.65m, 8.48°	2.11m, 8.38°	1.52m, 6.68°	**1.49m**, 3.43°	1.57m, 3.32°	2.12m, 2.97°	1.60m, **2.94°**
Street	-	-	-	20.7m, 25.7°	20.3m, 25.5°	19.6m, 19.9°	**19.0m**, **19.4°**
Chess	0.32m, 6.60°	0.37m, 7.24°	0.24m, 5.77°	0.24m, 5.77°	0.13m, 4.48°	0.26m, 6.34°	**0.10m**, **3.58°**
Fire	0.47m, 14.0°	0.43m, 13.7°	0.34m, 11.9°	0.27m, 11.8°	0.27m, 11.3°	0.28m, 10.3°	**0.15m**, **6.30°**
Heads	0.30m, 12.2°	0.31m, 12.0°	0.21m, 13.7°	0.18m, 12.1°	0.17m, 13.0°	0.17m, **7.98°**	**0.12m**, 8.15°
Office	0.48m, 7.24°	0.48m, 8.04°	0.30m, 8.08°	0.20m, 5.77°	0.19m, 5.55°	0.26m, 7.23°	**0.14m**, **3.11°**
Pumpkin	0.49m, 8.12°	0.61m, 7.08°	0.33m, 7.00°	0.25m, 4.82°	0.26m, 4.75°	0.22m, 5.18°	**0.17m**, **3.84°**
Red Kitchen	0.58m, 8.34°	0.58m, 7.54°	0.37m, 8.83°	0.24m, 5.52°	0.23m, 5.35°	0.55m, 16.7°	**0.15m**, **3.76°**
Stairs	0.48m, 13.1°	0.48m, 13.1°	0.40m, 13.7°	0.37m, 10.6°	0.35m, 12.4°	**0.17m**, 12.0°	0.19m, **8.30°**

Table 3. Ablation study with different backbones for selected dataset. Results superior to the best PoseNet strategy are in bold.

Scene	InceptionV3	VGG16 [69]	VGG19 [69]	ResNet50 [32]	ResNetV250 [33]	Xception [19]	DenseNet121 [38]	MobileNetV3 [37]	EfficientNetB0 [72]
Old Hospital	1.79m, **2.28°**	5.03m, 3.10°	1.93m, **1.70°**	13.11m, 13.96°	10.8m, 3.75°	2.21m,**3.11°**	1.98m, **2.12°**	12.14m, 11.10°	1.96m, **2.13°**
Shop	1.19m, **3.43°**	5.26m, 15.60°	4.87m, 14.01°	5.70m, 23.1°	5.20m, 6.20°	1.23m, **3.53°**	3.12m, **2.49°**	7.72m, 18.3°	4.15m, 6.23°
St. Mary's	**1.60m**, **2.94°**	2.78m, 4.80°	2.45m, 4.68°	1.72m, **3.28°**	**1.40m**, 3.41°	1.95m, 4.41°	2.01m, 3.95°	16.8m, 31.6°	6.24m, 7.28°
Chess	**0.10m**, **3.58°**	**0.11m**, **2.67°**	**0.13m**, **3.69°**	**0.07m**, **2.66°**	**0.10m**, **3.41°**	**0.095m**, **3.28°**	**0.080m**, **2.62°**	0.39m, 15.1°	**0.39m**, **15.1°**
Fire	**0.15m**, **6.30°**	0.28m, **8.69°**	**0.14m**, **9.29°**	0.39m, 20.1°	**0.22m**, **6.88°**	**0.21m**, **6.62°**	**0.16m**, **6.62°**	0.46m, 33.6°	**0.21m**, **7.60°**
Heads	**0.12m**, **8.15°**	0.20m, **11.9°**	0.20m, **10.4°**	0.29m, 15.6°	0.25m, 14.1°	0.22m, **12.9°**	**0.14m**, **8.90°**	0.25m, **12.5°**	0.17m, **9.33°**
Pumpkin	**0.17m**, **3.84°**	**0.22m**, **3.74°**	**0.22m**, **4.67°**	**0.10m**, **2.78°**	**0.14m**, **4.01°**	**0.18m**, **4.25°**	**0.13m**, **3.48°**	**0.22m**, **4.56°**	**0.13m**, **3.58°**
Red Kitchen	**0.15m**, **3.76°**	**0.02m**, **0.45°**	**0.02m**, **0.43°**	**0.01m**, **0.45°**	**0.02m**, **0.41°**	**0.02m**, **0.53°**	**0.02m**, **0.60°**	0.70m, 18.64°	**0.08m**, **1.92°**
Stairs	**0.19m**, **8.30°**	**0.22m**, **5.73°**	**0.26m**, **5.41°**	0.37m, **9.37°**	**0.25m**, **6.65°**	**0.19m**, **6.32°**	**0.22m**, **7.06°**	0.35m, **9.15°**	0.37m, **6.67°**

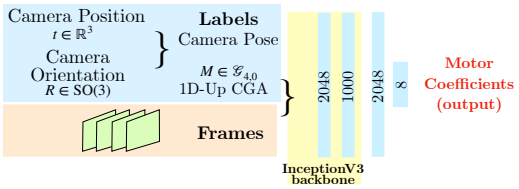


Figure 2. The original CGAPoseNet pipeline, which is geometry-agnostic and adds a significant number of parameters compared to baseline PoseNet.

13 different datasets. The indoor dataset, *7 Scenes*, was introduced in [68] and it includes *Chess, Fire, Heads, Office, Pumpkin, Red Kitchen* and *Stairs* datasets. The videos have been recorded with a Kinect RGB-D sensor and they all span a volume of less than $20m^3$.

The outdoor dataset, *Cambridge Landmarks*, was first presented along with PoseNet in [43]. It includes 6 datasets (*Great Court, King's College, Old Hospital, Street, Shop Facade, St. Mary's Church* and *Street*). The variability between each scene is significant, with *Shop Facade* spanning an area of 875m^2 and *Street* covering 50000m^2. The similarity between train and test set is also variable.

Each dataset includes RGB images extracted from the scene and labels of the position $\mathbf{t} = [x, y, z]$ and orientation $\mathbf{R}$ of the camera, expressed as either rotation matrices or quaternions, given an arbitrary reference frame. We converted labels into motors M and picked the curvature of the space λ (see Equation 5) to be proportional to the area spanned by the scene as described in [60].

4.2. Error metrics

Given a predicted motor $\hat{M}$ and a ground truth motor M, we measure the quality of the predicted pose by decomposing motors into their rotation and translation components and measure (i) translation error and (ii) rotation error.

We followed the procedure described in [60] and decomposed the motor M into a translation vector $\mathbf{t} \in \mathbb{R}^3$, the translation component, and into a rotor $R \in \mathcal{G}_{3,0}$, the rotation component.

We define the translation error between original position $\mathbf{t}$ and predicted position $\hat{\mathbf{t}}$ as:

$$e_{\mathbf{t}} = \|\hat{\mathbf{t}} - \mathbf{t}\|_1 \tag{10}$$

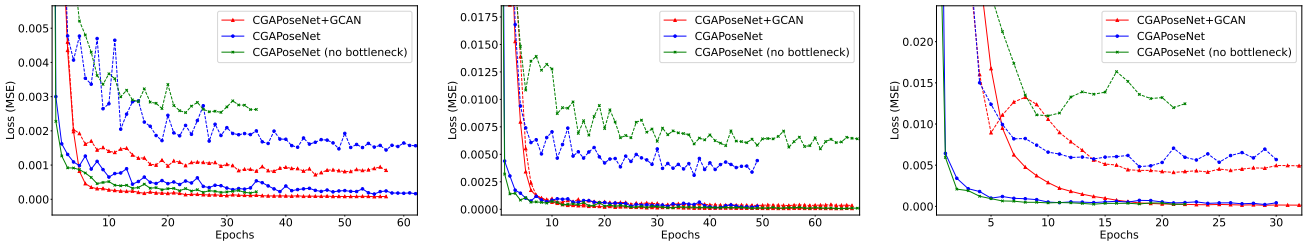
in a similar way described in [42, 44, 60]. The rotation error between a ground truth rotor R and predicted rotor $\hat{R}$ has been derived from [62, 82] and consistent with [60]. It is defined as:

$$e_R = \cos^{-1}(\langle R\tilde{\hat{R}}\rangle_0) \tag{11}$$

where $\langle\cdot\rangle_0$ denotes the component with grade 0, i.e. the scalar part of the geometric product. Since $R\tilde{\hat{R}} = 1$, when $\hat{R}$ is close to R the error goes to $0°$.

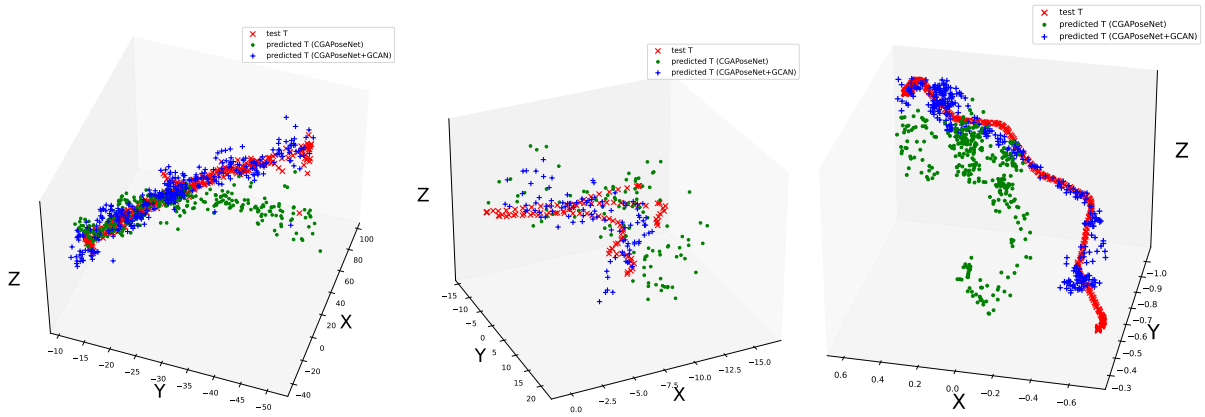
4.3. Training details

CGAPoseNet+GCAN has been trained in a supervised fashion with only RGB images I as inputs and camera poses expressed as motors M as labels. Weights are initialized starting from *ImageNet* [22, 43]. We employed an 80-20 train-validation split, a batch size of $B = 64$ and a number of epochs $E = 100$. Adam has been chosen as optimizer with exponentially decaying learning rate, with initial value



(a) *Chess* (b) *Fire* (c) *Heads*

Figure 3. Train (solid line) and validation (dashed line) losses for selected datasets. CGAPoseNet+GCAN attains the lowest loss profile due to its Geometric Clifford Algebra layers.



(a) *King's* (b) *Shop* (c) *Pumpkin*

Figure 4. Ground truth and predicted translation component of the pose over the test set for selected datasets.

$\eta = 10^{-4}$ and decay rate of 0.98. The rate of decay has been adjusted based on the training set dimension. To avoid overfitting, we implemented early stopping with patience $P = 12$ and restored the best weights based on the validation loss.

The training procedure adopted differs slightly between indoor and outdoor datasets. For *Cambridge Landmarks*, the network has been re-trained twice with decreasing starting learning rate, namely $\eta = \{10^{-4}, 10^{-5}, 10^{-6}\}$ and keeping the weights from the last training. For *7 Scenes* the network has been trained once.

The loss we minimize is

$$\mathcal{L} = \text{MSE}(M, \hat{M}) \tag{12}$$

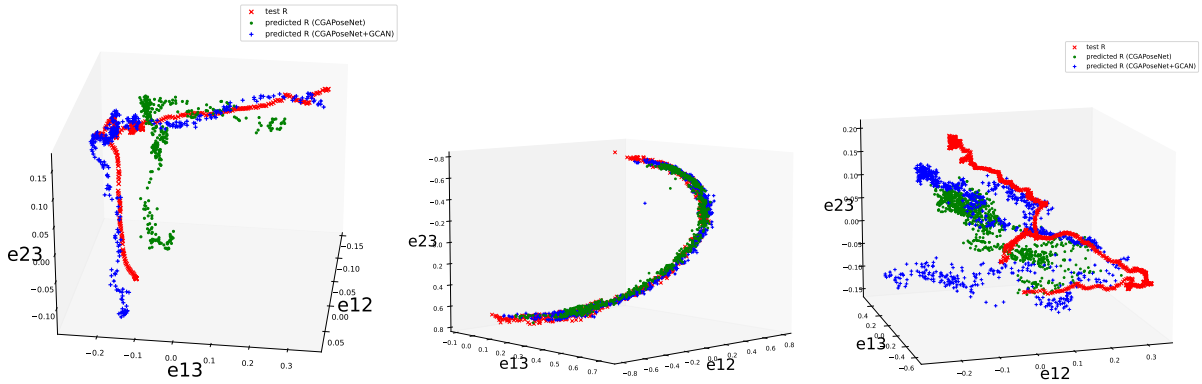
where $\hat{M}$ and M are the predicted and ground truth motors, as in [60]. The training time does not show noticeable difference with respect to the simple CGAPoseNet, both measured to be around $4s$/step.

The code is available in the form of two Jupyter notebooks, written on Google Colab Pro and run on a NVIDIA Tesla T4 GPU at 1.59 GHz. The backbone architecture has been implemented via the Keras API of TensorFlow, while the GCAN has been implemented via the TensorFlow Geometric Algebra library [39]. Operations in Geometric Algebra have been handled through Clifford [30]. Jupyter notebooks and output files are all available as supplementary materials.

5. Results

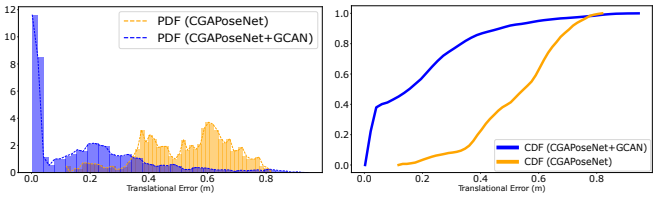
Results are summarized in Table 2. We report median translation and rotation errors, consistently with [41–44, 60], for 7 different approaches, namely 5 PoseNet approaches with different loss functions, CGAPoseNet and CGAPoseNet+GCAN (ours). Our approach signficantly reduces both errors by predicting a single mathematical object, the motor, through a geometry-aware network, presenting the lowest rotation error on 11 out of 13 datasets and the lowest translation error on 8 out of 13 datasets. For the *7 Scenes* dataset, even mean errors via CGAPoseNet+GCAN are below the others reported in Table 2.

To verify that the improvement comes indeed from the GCAN layers, we report training and validation losses in Figure 3. We compare CGAPoseNet, CGAPoseNet+GCAN and CGAPoseNet without bottleneck, i.e. by removing the last classification layer in the backbone and only adding one dense layer with 8 neurons. We do this so that

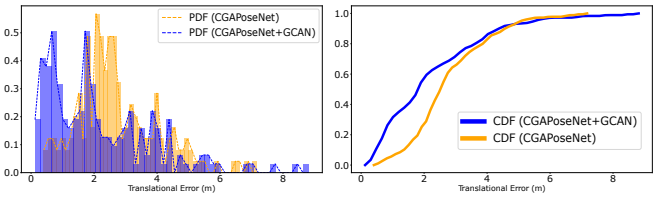


(a) *Pumpkin* (b) *Great Court* (c) *Red Kitchen*

Figure 5. Ground truth and predicted rotation component of the pose over the test set for selected datasets.

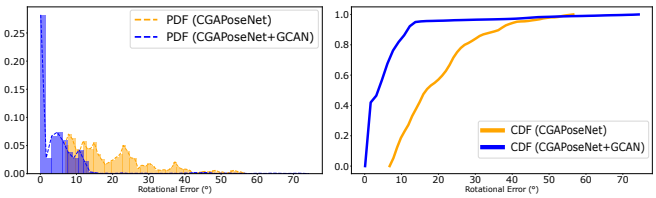


(a) PDF (left) and CDF (right) for the *Red Kitchen* dataset

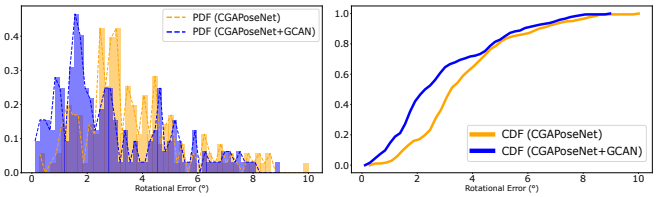


(b) PDF (left) and CDF (right) for the *Old Hospital* dataset

Figure 6. Translation error over the test set for selected datasets.



(a) PDF (left) and CDF (right) for the *Red Kitchen* dataset



(b) PDF (left) and CDF (right) for the *Old Hospital* dataset

Figure 7. Rotation error over the test set for selected datasets.

CGAPoseNet without bottleneck and CGAPoseNet+GCAN share the same backbone structure. It can be seen how the validation loss for CGAPoseNet+GCAN is lower than the other two approaches despite similar training loss profiles, showing (i) that our network generalizes better and (ii) that this is due to the GCAN layers and not to the backbone structure. CGAPoseNet without bottleneck performs worse than standard CGAPoseNet, justifying its structure.

In Figures 4-5 we display ground truth and predicted translation and rotation components, respectively, after breaking M down into $\mathbf{t} \in \mathbb{R}^3$ and $R \in \mathcal{G}_{4,0}$. In Figure 5 we plot the bivector components of the rotor R. CGAPoseNet+GCAN shows more accurate predictions, especially on previously unseen areas of the scene compared to CGAPoseNet (see Figure 5a). The improvement on the rotation component is less evident for outdoor datasets (see Figure 5b), but much more visible on indoor datasets (see Figure 5a-5c).

A comparison of the error distributions of the predictions via CGAPoseNet and CGAPoseNet+GCAN is given in Figures 6-7 for the translation and the rotation error, respectively: both errors are noticeably reduced with our geometry-aware approach in terms of both probability density function (PDF) and cumulative density function (CDF).

We visualize the outputs of the GCAN layer in Figure 8. As the GCAN works exclusively with motors, it is possible to interpret the intermediate layer outputs from a geometrical point of view as poses. The motor proposals (in yellow) are downsampled into progressively fewer poses until converging to the final prediction. Note the difference in scale (as also shown in Figure 1) between the outputs and how they cover progressively smaller areas. This explains why results are superior on the *7 Scenes* dataset, since the volume of Euclidean space to cover is significantly smaller. The curvature of the poses shows that we are working in the spherical space $\mathcal{G}_{4,0}$.

It is worth mentioning that the choice of the activation function influences the area in which poses are distributed, hence affecting how convergence is reached. In Figure 9,

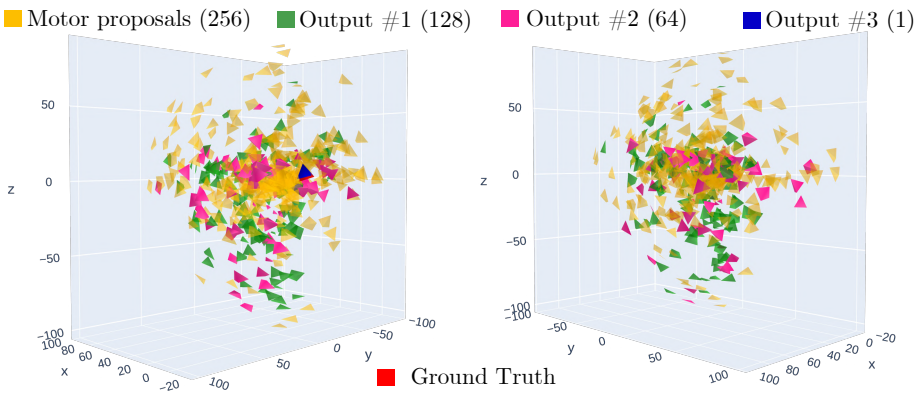


Figure 8. Input and output poses of the GCAN layers for a test image in the *Old Hospital* dataset (*relu* activation).

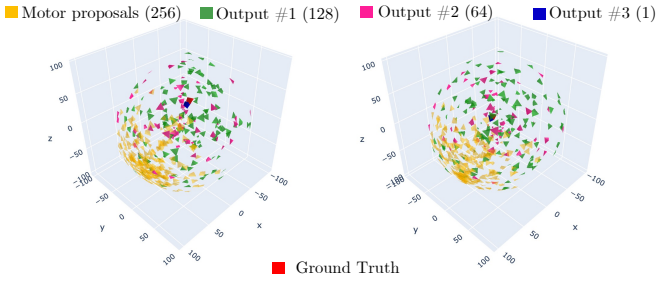


Figure 9. Input and output poses of the GCAN layers for a test image in the *Old Hospital* dataset (*tanh* activation).

for example, outputs obtained with a *tanh* activation function are presented. Predicted poses occupy a hemisphere in Figure 8, but they cover a full sphere in Figure 9: the intepretability of the GCANs intermediate ouputs allows us to design networks that minimize the loss function via different paths, meaning that we can customize the network prediction strategy based on the geometry of problem that we are trying to solve.

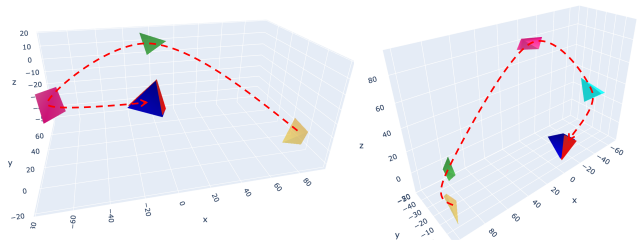


Figure 10. Average input and output poses of the GCAN layers for a test image in the *Old Hospital* dataset (*tanh* activation) with 3 (left) and 4 (right) sandwich product layers in the GCAN.

In Figure 10 the average pose in input and output of each GCAN layer is shown. We plot average poses with and without an additional layer with 32 neurons (output coloured in cyan). As geometric transformations are applied to the input poses by the sandwich product dense layers, the average pose converges towards ground truth, following clear traces in 3D space.

Lastly, an ablation study with different backbones has been performed, and results are reported in Table 3. Regardless of the backbone employed, CGAPoseNet+GCAN still outperforms the best performing PoseNet strategy in most indoor cases. For outdoor cases, we believe that backbones whose outputs are reshaped into more proposals need to be paired with GCAN layers with more units, to avoid bottlenecks from fast downsampling and to make sure that the volume covered by the scene is thoroughly explored.

6. Conclusion

We introduced CGAPoseNet+GCAN, an architecture to predict camera poses from images which expands CGAPoseNet with a network that works with Clifford Geometric Algebra objects. CGAPoseNet+GCAN is a geometric deep learning approach that allows for geometry-aware predictions of the camera poses. The backbone now only predicts *proposals* of suitable poses, on which geometric transformations are applied via GCAN until the optimal pose is found. The GCAN has been implemented through sandwich product layers that preserve objects' grades and whose outputs are fully interpretable geometrically. CGAPoseNet+GCAN, by working in the same mathematical space in which the predictions sit, significantly improves both baseline CGAPoseNet and PoseNet, reducing the pose regression error without requiring extra information about the captured scene and achieving state-of-the-art results. This is done by reducing the number of trainable parameters of CGAPoseNet by 17% and at no additional computational cost. We believe that GCANs have the potential to simplify and improve several computer vision approaches that have been solved through DL but without geometric information of the scene.

References

[1] Sameer Agarwal, Yasutaka Furukawa, Noah Snavely, Ian Simon, Brian Curless, Steven M Seitz, and Richard Szeliski. Building rome in a day. *Communications of the ACM*, 54(10):105–112, 2011. 1

[2] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359, 2008. 1

[3] Eduardo Bayro-Corrochano, Luis Lechuga-Gutiérrez, and Marcela Garza-Burgos. Geometric techniques for robotics and hmi: Interpolation and haptics in conformal geometric algebra and control using quaternion spike neural networks. *Robotics and Autonomous Systems*, 104:72–84, 2018. 3

[4] Eduardo Jose Bayro-Corrochano. Geometric neural computing. *IEEE Transactions on Neural Networks*, 12(5):968–986, 2001. 3

[5] Michael Bloesch, Jan Czarnowski, Ronald Clark, Stefan Leutenegger, and Andrew J Davison. Codeslam—learning a compact, optimisable representation for dense visual slam. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2560–2568, 2018. 1

[6] Eric Brachmann, Alexander Krull, Frank Michel, Stefan Gumhold, Jamie Shotton, and Carsten Rother. Learning 6d object pose estimation using 3d object coordinates. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part II 13*, pages 536–551. Springer, 2014. 2

[7] Eric Brachmann, Frank Michel, Alexander Krull, Michael Ying Yang, Stefan Gumhold, et al. Uncertainty-driven 6d pose estimation of objects and scenes from a single rgb image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3364–3372, 2016. 2

[8] Johannes Brandstetter, Rianne van den Berg, Max Welling, and Jayesh K Gupta. Clifford neural layers for pde modeling. *arXiv preprint arXiv:2209.04934*, 2022. 3

[9] Romain Brégier. Deep regression on manifolds: a 3d rotation case study. In *2021 International Conference on 3D Vision (3DV)*, pages 166–174. IEEE, 2021. 3

[10] Johann Brehmer, Pim de Haan, Sönke Behrends, and Taco Cohen. Geometric algebra transformers. *arXiv preprint arXiv:2305.18415*, 2023. 3

[11] Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017. 2

[12] Sven Buchholz. Quaternionic spinor mlp. In *Proc. European Symposium on Artificial Neural Networks, 2000*, pages 377–382, 2000. 3

[13] Sven Buchholz and Nicolas Le Bihan. Polarized signal classification by complex and quaternionic multi-layer perceptrons. *International journal of neural systems*, 18(02):75–85, 2008. 3

[14] Sven Buchholz and Gerald Sommer. Clifford algebra multilayer perceptrons. *Geometric Computing with Clifford Algebras: Theoretical Foundations and Applications in Computer Vision and Robotics*, pages 315–334, 2001. 3

[15] Sven Buchholz and Gerald Sommer. On clifford neurons and clifford multi-layer perceptrons. *Neural Networks*, 21(7):925–935, 2008. 3

[16] Sven Buchholz, Kanta Tachibana, and Eckhard MS Hitzer. Optimal learning rates for clifford neurons. In *Artificial Neural Networks–ICANN 2007: 17th International Conference, Porto, Portugal, September 9-13, 2007, Proceedings, Part I 17*, pages 864–873. Springer, 2007. 3

[17] Wenming Cao, Canta Zheng, Zhiyue Yan, and Weixin Xie. Geometric deep learning: progress, applications and challenges. *Science China Information Sciences*, 65(2):126101, 2022. 3

[18] Jiayi Chen, Yingda Yin, Tolga Birdal, Baoquan Chen, Leonidas J Guibas, and He Wang. Projective manifold gradient layer for deep rotation regression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6646–6655, 2022. 3

[19] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258, 2017. 5

[20] Pieter Chys. Application of geometric algebra for the description of polymer conformations. *The Journal of chemical physics*, 128(10):104107, 2008. 2

[21] E Bayro Corrochano, Sven Buchholz, and Gerald Sommer. Selforganizing clifford neural network. In *Proceedings of International Conference on Neural Networks (ICNN'96)*, volume 1, pages 120–125. IEEE, 1996. 3

[22] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 5

[23] Chris Doran and Anthony Lasenby. *Geometric algebra for physicists*. Cambridge University Press, 2003. 2

[24] Ahmed Elmoogy, Xiaodai Dong, Tao Lu, Robert Westendorp, and Kishore Reddy. Pose-gnn: Camera pose estimation system using graph neural networks. *arXiv preprint arXiv:2103.09435*, 2021. 3

[25] Jakob Engel, Thomas Schöps, and Daniel Cremers. Lsd-slam: Large-scale direct monocular slam. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part II 13*, pages 834–849. Springer, 2014. 1

[26] Matthias Fey and Jan Eric Lenssen. Fast graph representation learning with pytorch geometric. *arXiv preprint arXiv:1903.02428*, 2019. 3

[27] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981. 2

[28] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE transactions on pattern analysis and machine intelligence*, 32(8):1362–1376, 2009. 2

[29] Charles G Gunn and Steven De Keninck. Geometric algebra and computer graphics. In *ACM SIGGRAPH 2019 Courses*, pages 1–140. 2019. 2

[30] Hugo Hadfield, Eric Wieser, Alex Arsenovic, Robert Kern, and The Pygae Team. pygae/clifford. 6

[31] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003. 2

[32] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 5

[33] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*, pages 630–645. Springer, 2016. 5

[34] David Held, Sebastian Thrun, and Silvio Savarese. Learning to track at 100 fps with deep regression networks. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, pages 749–765. Springer, 2016. 1

[35] David Hestenes. *Space-time algebra*. Springer, 2015. 2

[36] Eckhard Hitzer. Introduction to clifford's geometric algebra. *Journal of the Society of Instrument and Control Engineers*, 51(4):338–350, 2012. 2

[37] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017. 5

[38] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017. 5

[39] Robin Kahlow. Tensorflow geometric algebra. 6

[40] Abhishek Kar, Shubham Tulsiani, Joao Carreira, and Jitendra Malik. Category-specific object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1966–1974, 2015. 1

[41] Alex Kendall and Roberto Cipolla. Modelling uncertainty in deep learning for camera relocalization. In *2016 IEEE international conference on Robotics and Automation (ICRA)*, pages 4762–4769. IEEE, 2016. 2, 4, 5, 6

[42] Alex Kendall and Roberto Cipolla. Geometric loss functions for camera pose regression with deep learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5974–5983, 2017. 2, 3, 4, 5, 6

[43] Alex Kendall, Matthew Grimes, and Roberto Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *Proceedings of the IEEE international conference on computer vision*, pages 2938–2946, 2015. 2, 3, 4, 5, 6

[44] Alex Guy Kendall. *Geometry and uncertainty in deep learning for computer vision*. PhD thesis, University of Cambridge, UK, 2019. 5, 6

[45] Georg Klein and David Murray. Parallel tracking and mapping for small ar workspaces. In *2007 6th IEEE and ACM international symposium on mixed and augmented reality*, pages 225–234. IEEE, 2007. 1

[46] Yasuaki Kuroe. Models of clifford recurrent neural networks and their dynamics. In *The 2011 international joint conference on neural networks*, pages 1035–1041. IEEE, 2011. 3

[47] Anthony Lasenby. Recent applications of conformal geometric algebra. In *Computer Algebra and Geometric Algebra with Applications*, pages 298–328. Springer, 2004. 3

[48] Anthony Lasenby. Rigid body dynamics in a constant curvature space and the '1d-up'approach to conformal geometric algebra. In *Guide to geometric algebra in practice*, pages 371–389. Springer, 2011. 3

[49] Anthony Lasenby. A 1d up approach to conformal geometric algebra: applications in line fitting and quantum mechanics. *Advances in Applied Clifford Algebras*, 30(2):1–16, 2020. 3

[50] Joan Lasenby and Leo Dorst. *Guide to geometric algebra in practice*. Springer, 2011. 2

[51] Carlile Lavor and Rafael Alves. Oriented conformal geometric algebra and the molecular distance geometry problem. *Advances in Applied Clifford Algebras*, 29(1):1–15, 2019. 2

[52] Yanping Li, Yue Wang, Rui Wang, Yi Wang, Kaili Wang, Xiangyang Wang, Wenming Cao, and Wei Xiang. Ga-cnn: Convolutional neural network based on geometric algebra for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 60:1–14, 2022. 3

[53] Qifan Liu and Wenming Cao. Geometric algebra graph neural network for cross-domain few-shot classification. *Applied Intelligence*, 52(11):12422–12435, 2022. 3

[54] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60:91–110, 2004. 1

[55] Xiankai Lu, Chao Ma, Bingbing Ni, Xiaokang Yang, Ian Reid, and Ming-Hsuan Yang. Deep regression tracking with shrinkage loss. In *Proceedings of the European conference on computer vision (ECCV)*, pages 353–369, 2018. 1

[56] Eric Marchand, Hideaki Uchiyama, and Fabien Spindler. Pose estimation for augmented reality: a hands-on survey. *IEEE transactions on visualization and computer graphics*, 22(12):2633–2651, 2015. 1

[57] Iaroslav Melekhov, Juha Ylioinas, Juho Kannala, and Esa Rahtu. Relative camera pose estimation using convolutional neural networks. In *Advanced Concepts for Intelligent Vision Systems: 18th International Conference, ACIVS 2017, Antwerp, Belgium, September 18-21, 2017, Proceedings 18*, pages 675–687. Springer, 2017. 2

[58] Federico Monti, Davide Boscaini, Jonathan Masci, Emanuele Rodola, Jan Svoboda, and Michael M Bronstein. Geometric deep learning on graphs and manifolds using mixture model cnns. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5115–5124, 2017. 3

[59] JK Pearson and DL Bisset. Neural networks in the clifford domain. In *Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN'94)*, volume 3, pages 1465–1469. IEEE, 1994. 3

[60] Alberto Pepe and Joan Lasenby. Cga-posenet: Camera pose regression via a 1d-up approach to conformal geometric algebra. *arXiv preprint arXiv:2302.05211*, 2023. 2, 3, 4, 5, 6

[61] Alberto Pepe and Joan Lasenby. Modeling orientational features via geometric algebra for 3d protein coordinates prediction. *Authorea Preprints*, 2023. 2

[62] Alberto Pepe, Joan Lasenby, and Pablo Chacón. Learning rotations. Conference on Applied Geometric Algebras in Computer Science and Engineering (AGACSE), 2021. 3, 5

[63] Alberto Pepe, Joan Lasenby, and Pablo Chacon. Using a graph transformer network to predict 3d coordinates of proteins via geometric algebra modelling. Technical report, EasyChair, 2022. 2

[64] Valentin Peretroukhin, Matthew Giamou, David M Rosen, W Nicholas Greene, Nicholas Roy, and Jonathan Kelly. A smooth representation of belief over so (3) for deep rotation learning with uncertainty. *arXiv preprint arXiv:2006.01031*, 2020. 3

[65] David Ruhe, Johannes Brandstetter, and Patrick Forré. Clifford group equivariant neural networks. *arXiv preprint arXiv:2305.11141*, 2023. 3

[66] David Ruhe, Jayesh K Gupta, Steven de Keninck, Max Welling, and Johannes Brandstetter. Geometric clifford algebra networks. *arXiv preprint arXiv:2302.06594*, 2023. 2, 3

[67] Ashutosh Saxena, Justin Driemeyer, and Andrew Y Ng. Learning 3-d object orientation from images. In *2009 IEEE International conference on robotics and automation*, pages 794–800. IEEE, 2009. 3

[68] Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon. Scene coordinate regression forests for camera relocalization in rgb-d images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2930–2937, 2013. 5

[69] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 5

[70] Shyamal Somaroo, Anthony Lasenby, and Chris Doran. Geometric algebra and the causal approach to multiparticle quantum mechanics. *Journal of Mathematical Physics*, 40(7):3327–3340, 1999. 2

[71] Takafumi Taketomi, Kazuya Okada, Goshiro Yamamoto, Jun Miyazaki, and Hirokazu Kato. Camera pose estimation under dynamic intrinsic parameter change for augmented reality. *Computers & graphics*, 44:11–19, 2014. 1

[72] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019. 5

[73] Keisuke Tateno, Federico Tombari, Iro Laina, and Nassir Navab. Cnn-slam: Real-time dense monocular slam with learned depth prediction. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6243–6252, 2017. 1

[74] Bill Triggs, Philip F McLauchlan, Richard I Hartley, and Andrew W Fitzgibbon. Bundle adjustment—a modern synthesis. In *Vision Algorithms: Theory and Practice: International Workshop on Vision Algorithms Corfu, Greece, September 21–22, 1999 Proceedings*, pages 298–372. Springer, 2000. 2

[75] Jack Valmadre, Luca Bertinetto, Joao Henriques, Andrea Vedaldi, and Philip HS Torr. End-to-end representation learning for correlation filter based tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2805–2813, 2017. 1

[76] Florian Walch, Caner Hazirbas, Laura Leal-Taixe, Torsten Sattler, Sebastian Hilsenbeck, and Daniel Cremers. Image-based localization using lstms for structured feature correlation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 627–637, 2017. 3, 5

[77] Rui Wang, Miaomiao Shen, Xiangyang Wang, and Wenming Cao. Rga-cnns: Convolutional neural networks based on reduced geometric algebra. *Sci. China Inf. Sci*, 64(129101):1–129101, 2021. 3

[78] Rich Wareham, Jonathan Cameron, and Joan Lasenby. Applications of conformal geometric algebra in computer vision and graphics. In *International Workshop on Mathematics Mechanization*, pages 329–349. Springer, 2004. 2

[79] Sitao Xiang and Hao Li. Revisiting the continuity of rotation representations in neural networks. *arXiv preprint arXiv:2006.06234*, 2020. 3

[80] Jian Ye, Jiangqun Ni, and Yang Yi. Deep learning hierarchical representations for image steganalysis. *IEEE Transactions on Information Forensics and Security*, 12(11):2545–2557, 2017. 2

[81] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph neural networks: A review of methods and applications. *AI open*, 1:57–81, 2020. 3

[82] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 5745–5753, 2019. 3, 5

[83] Yi Zhou, Guillermo Gallego, Henri Rebecq, Laurent Kneip, Hongdong Li, and Davide Scaramuzza. Semi-dense 3d reconstruction with a stereo event camera. In *Proceedings of the European conference on computer vision (ECCV)*, pages 235–251, 2018. 1

[84] Jingwen Zhu and Jitao Sun. Global exponential stability of clifford-valued recurrent neural networks. *Neurocomputing*, 173:685–689, 2016. 3