FIGURE: Form, Invariance, and Geometric Understanding for Representation Evaluation

Erik J. Bekkers

February 26, 2025

1 Introduction

This document provides details on the FIGURE dataset as hosted on the repository https://github.com/ebekkers/FIGURE. Some experiments on this dataset are already reported in its `README.md` file. We propose a synthetic dataset of figures for controlled experiments in robustness evaluation. The dataset consists of point clouds in the Lie group SE(2), with each figure represented as a tree of relative group actions. The primary objective is to assess robustness against texture bias by disentangling geomeric pose information from appearance features.

2 Mathematical Formulation

Each figure is modeled as a set of transformations in SE(2), describing the global pose and limb articulations.

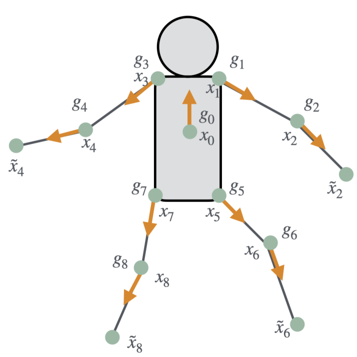

Figure 1: Structure of the synthetic stick figure. Each node g_i represents a transformation/pose in SE(2). The root pose g_0 is oriented upward ($\theta_0 = 90°$), and each limb extends hierarchically. Each joint has an associated position x_i, and we denote the end points of each limb with a tilde ($\tilde{}$), e.g., $\tilde{x}_2$ denotes the end point of the lower left arm that starts at location x_2.

2.1 Body Representation

The figure's global pose is defined as:

$$g_0 = (x_0, \theta_0)\,, \quad x_0 \in \mathbb{R}^2\,, \quad \theta_0 \in S^1 \equiv SO(2)\,, \tag{1}$$

where x_0 represents the central part of the torso, and θ_0 indicates the figure's orientation. We use the convention that $\theta = 0$ is aligned with the horizontal axis, and θ rotates counter-clockwise such that $\theta_0 = 90°$ means the figure is standing upright. In this document we provide the angles in degrees (°). Each limb's pose is defined recursively as:

$$g_i = g_{i-1} \cdot \Delta g_i, \quad \Delta g_i \in \mathrm{SE}(2) \tag{2}$$

where Δg_i is a relative transformation describing the spatial relation between consecutive limbs.

2.2 Body Joints

Each limb's pose $g_i = (x_i, \theta_i) \in \mathrm{SE}(2)$ consists of a position $x_i \in \mathbb{R}^2$ and an orientation θ_i. We define the projection operator $\pi : SE(2) \to \mathbb{R}^2$:

$$\pi(g_i) = x_i \tag{3}$$

which extracts the position component from the group element, and essentially distills the *joint* location.

For each limb, we also define an *end-point* $\tilde{x}_i$, which corresponds to the extremity of the limb segment. Instead of first generating a new group element $\tilde{g}_i$, we compute the end-point position directly as:

$$\tilde{x}_i = g_i \cdot \Delta\tilde{x}_i \tag{4}$$

where we overload the symbol $\cdot$ to now also represents the group action on $\mathbb{R}^2$. The spatial offset $\Delta\tilde{x}_i$ is given by:

$$\Delta\tilde{x}_i = \begin{bmatrix} \ell^{\text{arm}} \\ 0 \end{bmatrix} \quad \text{or} \quad \Delta\tilde{x}_i = \begin{bmatrix} \ell^{\text{leg}} \\ 0 \end{bmatrix} \tag{5}$$

depending on whether i represents an arm or a leg.

2.3 Pose Definitions

Each relative limb transformation Δg_i consists of a translation and a rotation. The pose transformations for different limbs are given in Table. 2.3.

The joint angles θ_i are drawn from probability distributions that include both *natural body variations* and *class-specific variations*, where we define *four classes*: both arms up (up-up), both arms down (down-down), left arm up and right arm down (up-down), and right arm up and left arm down (down-up). See Table. 2.3 for the random variables. Here, $p(\theta_1|\text{up})$ and $p(\theta_3|\text{up-up})$ describe the angle distributions when the figure belongs to the up-up class, whereas $p(\theta_1|\text{down})$ and $p(\theta_3|\text{down-down})$ describe the angle distributions for the down-down class. The remaining joint distributions capture natural variability in body posture (small angle variations at the joints), independent of class labels.

Table 1: Pose transformations and corresponding relative transformations for each limb.

Limb	**Pose Transformation**	**Relative Transformation**
Left upper arm	$g_1 = g_0 \cdot \Delta g_1$	$\Delta g_1 = (h/2, -w/2, \theta_1)$
Left lower arm	$g_2 = g_1 \cdot \Delta g_2$	$\Delta g_2 = (\ell^{\text{arm}}, 0, \theta_2)$
Right upper arm	$g_3 = g_0 \cdot \Delta g_3$	$\Delta g_3 = (h/2, w/2, \theta_3)$
Right lower arm	$g_4 = g_3 \cdot \Delta g_4$	$\Delta g_4 = (\ell^{\text{arm}}, 0, \theta_4)$
Left upper leg	$g_5 = g_0 \cdot \Delta g_5$	$\Delta g_5 = (-h/2, -w/2, \theta_5)$
Left lower leg	$g_6 = g_5 \cdot \Delta g_6$	$\Delta g_6 = (\ell^{\text{leg}}, 0, \theta_6)$
Right upper leg	$g_7 = g_0 \cdot \Delta g_7$	$\Delta g_7 = (-h/2, w/2, \theta_7)$
Right lower leg	$g_8 = g_7 \cdot \Delta g_8$	$\Delta g_8 = (\ell^{\text{leg}}, 0, \theta_8)$

3 The Shape class and Torch dataloader

The repository contains two important files.

3.1 The Shape class

In `figure_class.py` you'll find a python class `Shape` (based on `numpy`) from which you can sample random figures. This class has the methods `.resample(g, pose_class)` which randomly samples the keypoints according to the distribution of Table. 2.3 given the specified `pose_class` which takes the options `up-up`, `down-down`, `up-down`, `down-up` and applies a group action by g. Setting `g=(0,0,0)` generates the figure in up-right position at the center of the image.

The class also has the method `.set_pose_from_landmarks(points_r2, resolution)` which sets all the variables from a given set of 13 keypoints. This may be useful in case you want to visualze the result of keypoint detection method, or a point cloud generator.

Then there are 2 visualization methods. `.visualize()` generates a matplotlib figure. `.render(image_ize)` generates a numpy array image of the figure, and also returns the 13 keypoints in pixel coordinates, as well as the 9 $SE(2)$ elements in pixel coordinates. This is the function used to generated the training data provided by the dataloader.

3.2 The Torch dataloader

In `data_loader_torch.py` you'll find the torch dataset `FigureDataset`, which inherits from `torch.utils.data.Dataset`. It returns 4 items: the image, the

Table 2: Random variable distributions for limb orientations and body proportions. Class-specific variations are explicitly indicated.

Random Variable	=	**Distribution**
$p(\theta_1 \mid \text{up-up})$	=	$U(-85, -10)$
$p(\theta_1 \mid \text{up-down})$	=	$U(-85, -10)$
$p(\theta_1 \mid \text{down-down})$	=	$U(-170, -95)$
$p(\theta_1 \mid \text{down-up})$	=	$U(-170, -95)$
$p(\theta_2)$	=	$U(-10, 10)$
$p(\theta_3 \mid \text{up-up})$	=	$U(10, 85)$
$p(\theta_3 \mid \text{up-down})$	=	$U(95, 170)$
$p(\theta_3 \mid \text{down-down})$	=	$U(95, 170)$
$p(\theta_3 \mid \text{down-up})$	=	$U(10, 85)$
$p(\theta_4)$	=	$U(-10, 10)$
$p(\theta_5)$	=	$U(170, 225)$
$p(\theta_6)$	=	$U(-10, 10)$
$p(\theta_7)$	=	$U(135, 190)$
$p(\theta_8)$	=	$U(-10, 10)$
$p(w)$	=	$U(w_{\min}, w_{\max})$
$p(h)$	=	$U(h_{\min}, h_{\max})$
$p(\ell^{\text{arm}})$	=	$U(\ell^{\text{arm}}_{\min}, \ell^{\text{arm}}_{\max})$
$p(\ell^{\text{leg}})$	=	$U(\ell^{\text{leg}}_{\min}, \ell^{\text{leg}}_{\max})$

13 keypoints in pixel coordinates (array indices), the 9 $SE(2)$ keypoints, and the class label.

It is also possible to make the shirt color a random variable by setting the parameter `torso_colors`, `color_probabilities`, `color_consistency`. We use this to create biases in the dataset by correlating class labels with a certain color preference (as reflected by the color probabilities). See the `README.md` for more info.