

Koopman pose predictions for temporally consistent human walking estimations

Marc Mitjans¹, David M. Levine², Louis N. Awad³, Roberto Tron¹

Abstract—We tackle the problem of tracking the human lower body as an initial step toward an automatic motion assessment system for clinical mobility evaluation, using a multi-modal system that combines Inertial Measurement Unit (IMU) data, RGB images, and point cloud depth measurements. This system applies the factor graph representation to an optimization problem that provides 3-D skeleton joint estimations. In this paper, we focus on improving the temporal consistency of the estimated human trajectories to greatly extend the range of operability of the depth sensor. More specifically, we introduce a new factor graph factor based on Koopman theory that embeds the nonlinear dynamics of several lower-limb movement activities. This factor performs a two-step process: first, a custom activity recognition module based on spatial temporal graph convolutional networks recognizes the walking activity; then, a Koopman pose prediction of the subsequent skeleton is used as an *a priori* estimation to drive the optimization problem toward more consistent results. We tested the performance of this module a dataset composed of multiple clinical lower-limb mobility tests, and we show that our approach reduces outliers on the skeleton form by almost 1 m, while preserving natural walking trajectories at depths up to more than 10 m.

I. INTRODUCTION

It is undeniable that the process of aging comes with an increasing frailty and loss of mobility. However, although frail adults constitute the sickest, fastest growing, and most expensive segment of the US population [1], the US healthcare system still does not offer solutions to promptly detect the onset of frailty and act accordingly. Instead, it passively reacts to accidents that occur as a consequence of functional decline, which entail expensive and time-consuming hospital-based interventions. In this light, home-based therapies that monitor the motor skills of the patient on a continuous daily basis could allow therapists to make timely diagnoses of mobility decline, which would result in an increase of the patient's quality of life. For this purpose, we propose an improved low-cost multi-modal system for human motion tracking, aimed at facilitating mobility assessment in the home through various every-day activities. We build upon a preliminary version of the system [2]; such system uses a

factor graph formulation to combine inertial measurements from four Inertial Measurement Units (IMUs, attached to the lower limbs of the user) with RGB and depth images from an depth camera, to provide 3-D human movement estimations. In this paper, we expand the factor graph formulation with a novel factor that combines deep learning and Koopman theory to first predict the walking activity being performed, and then compute a prediction of the user motion to bias the factor graph optimization toward a stronger temporal consistency.

Related work. Our work on human pose estimation is derived from a fusion between recent techniques used for robotic localization, namely Visual-Inertial Odometry (VIO), and data-driven machine learning models. VIO techniques use a fusion of inertial measurements with visual information. There is also research in the literature on the use of IMUs, kinematic models and vision techniques for human pose prediction. For instance, the authors in [3] used IMU data and a kinematic model to track the upper limbs in neurorehabilitation exercises; the work from [4] showed that fusing IMU and vision data outperforms IMU-only approaches; and multiple studies, such as [5] and [6], have developed 3-D pose estimation algorithms with IMU, LiDAR and multiple-camera data.

With the recent rise of deep neural networks, there has also been an increasing interest in tracking the human pose and recognizing human activities from a deep learning perspective. For example, a common open-source library used for pose estimation is OpenPose [7], a Convolutional Neural Network (CNN)-based system for 2-D multi-human pose estimation from RGB images. The authors of [8] proposed recurrent-based CNNs to track the human body with spatial and temporal information on video sequences; the authors of [9], [10] and [11] suggested methods to infer the 3-D human joint positions from multiple 2-D images; and the research from [12] uses the Koopman operator to constrain the training of a deep autoencoder on silhouette images, whose feature space is then used for gait recognition. More directly related to the problem of human activity recognition, state-of-the-art results are based on Graph Neural Networks (GCNs), which treat the human skeleton joints as graph nodes, and their connections (bones) as graph edges. For instance, the authors in [13] proposed a module to select the most informative frames in a skeleton sequence and fuse it with a GCN module; the authors in [14] presented a GCN architecture that fuses information both from nodes and skeleton edges; and the authors in [15] introduced the spatial temporal GCN, which applies graph convolutions on the spatial domain and regular convolutions on the temporal domain.

¹Department of Mechanical Engineering, Boston University, 110 Commonwealth Mall, MA 02215, USA {mmitjans, tron}@bu.edu. Affiliates are supported by NIH R01AG067394-02. R. Tron is additionally supported by NSF NRI-1734454.

²Division of General Internal Medicine and Primary Care, Brigham and Women's Hospital, Harvard Medical School, 75 Francis St, Boston, MA 02115, USA {dmlevine}@bwh.harvard.edu.

³College of Health and Rehabilitation Sciences: Sargent College, Boston University, 635 Commonwealth, MA 02215, USA {louawad}@bu.edu. Affiliates are supported by AHA 18IPA34170487.

We thank Michail Theofanis, Aaron Horowitz and Jialan Sun for their invaluable help in labeling our datasets.

Paper contributions. Our system explores the capabilities of Koopman theory in the field of human pose estimation. We show that Koopman-based predictions of the human pose placed within the factor graph optimization loop condition the skeleton estimations towards more consistent gait estimations. This allows the system to obtain more realistic motion estimates even in the presence of significant noise (e.g., far from the camera), thus overcoming a main limitation in [2]. We analyze the performance of our system on a dataset of clinically-relevant lower limb mobility tests.

The remainder of the paper is organized as follows: Sec. II presents the preliminary concepts required to understand Sec. III, which describes the implementation of our proposed Koopman factor. Sec. IV presents the conducted experiments and the discussion of the results, and finally Sec. V summarizes the conclusions and introduces potential future work.

II. PRELIMINARIES

We begin by providing a brief overview of the optimization framework for human walking estimations presented in [2]. Then, we introduce the two new elements that constitute the core of this paper: the Koopman operator and the spatial-temporal graph neural network for walking activity selection.

A. Human kinematic model

We model the human kinematics with a directed tree, $H = (V, E)$. Each node $i \in V$ represents a joint of the articulated human model, and is associated to 3-D trajectories $\mathbf{n}_k^i \in \mathbb{R}^3$ expressed in an inertial reference frame, where k is a discrete time index associated to a keyframe time t_k . Likewise, $(i, j) \in E$ represents a link connecting joints i and j , and is associated to a time-invariant length $l^{ij} \in \mathbb{R}$, and a time-dependent extrinsic rotation $\mathbf{R}_k^{ij} \in SO(3)$, which represents the transformation from the inertial frame to a frame rigidly affixed to the link. By convention, the z -axis of each rotation is placed along its corresponding link pointing from parent node i to child node j . Given the 3-D coordinates of the root node \mathbf{n}_k^0 , this parameterization enables reconstructing the full human skeleton with the following kinematic relation:

$$\mathbf{n}_k^j = \mathbf{n}_k^i + l^{ij} \mathbf{R}_k^{ij} \mathbf{e}_3, \quad (1)$$

where $\mathbf{e}_3 = [0 \ 0 \ 1]^\top$ corresponds to the standard z Cartesian coordinate basis. Our human skeleton tree is composed of 7 joints (feet, knees, hips, and sternum) and 6 links, with the sternum acting as the root node.

We define the state of our human skeleton at time t_k by concatenating the root node position \mathbf{n}_k^0 with all the link rotations \mathbf{R}_k^{ij} to form state \mathcal{R}_k . Likewise, we concatenate all link lengths l^{ij} to form the set of parameters \mathcal{L} .

B. Factor graphs

Unrolling equation (1) for all joints and links across a sliding window of K keyframes, we obtain a graph structure. This motivates representing our gait estimation problem using the factor graph formulation. A factor graph [16] $G = (S, F)$ is a bipartite graph which represents a complex multivariate probabilistic model as a multiplication of simpler models

(factors). Each factor $f_k \in F$ for $k \in K$ depends on a small subset of the system variables (states) $s_k \in S$ and encodes the likelihood of s_k given certain external measurements $z_k \in Z$, $f_k(s_k; z_k)$. The joint probability $P(S|Z)$ of all the states $S = \{s_k\}$ given all the measurements $Z = \{z_k\}$ can then be expressed as

$$P(S|Z) \propto \prod_{k \in K} f_k(s_k; z_k). \quad (2)$$

The maximum likelihood estimation (MLE) of S is obtained by minimizing the negative log-likelihood of $P(S|Z)$. Assuming that factors f_k follow zero-mean Gaussian distributions, one can express the MLE of (2) as the least-squares solution of an optimization problem depending on residuals $\mathbf{r}_k(s_k) \in \mathbb{R}^n$ [17],

$$S^* = \arg \min -\ln P(S|Z) = \arg \min -\sum_{k \in K} \|\mathbf{r}_k\|_{\Sigma_k}^2, \quad (3)$$

where Σ_k corresponds to the noise covariance of the measurement z_k associated to the factor f_k . The residuals \mathbf{r}_k express the difference between the measurement z_k and the (possibly nonlinear) measurement function $h_k(s_k)$. This representation of the factor graph optimization problem intuitively defines Σ_k^{-1} as the *weight* associated to each residual \mathbf{r}_k . With this insight, we can use factors that are based on machine learning algorithms, treating their weights (covariance matrices) as design parameters.

The optimization problem (3) is generally solved using gradient descent or quasi-Newton methods; this requires computing the gradients of the residuals, which requires finding expressions for the Jacobian algorithms with respect to the optimization variables.

C. Visual-inertial 3-D human pose estimations

We previously presented a visual-inertial system for reconstructing 3-D human poses [2]; this system applies the factor graph formulation described in Sec. II-B to the kinematic model from Sec. II-A to estimate the 3-D coordinates of the root node \mathbf{n}_k^0 , the extrinsic rotations \mathbf{R}_k^{ij} , and the intrinsic link lengths l^{ij} , over a sliding window to approximate the 3-D skeleton movement. This is achieved by fusing information from four IMUs strapped on the two legs (two shanks and two thighs) with synchronized images and depth point clouds captured by an RGB-D camera. RGB images provide 2-D pixel coordinates of the human skeleton joints, while the point cloud provides their depth component. These measurement sources are encoded in four different types of factors over their corresponding subset of state variables: 1) The IMU factor, which computes the estimated rotation change of the links between adjacent keyframes by applying *preintegration* theory [18] on the angular velocities; 2) the Image factor, which computes the reprojection error between the xy coordinates of the predicted joints under perspective projection [19] and the pixel coordinates measured by OpenPose; 3) the Depth factor, which computes the depth error on the estimated joints; and 4) the Contact factor, which uses a logistic regressor on

IMU data to predict feet contacts and thus avoid estimating unrealistic skids (sliding of the feet).

This preliminary work was implemented using the GTSAM library for factor graph optimization [20], and it showed the potential of visual-inertial filtering and factor graphs applied to 3-D human pose estimation. However, the current technology for time-of-flight cameras leads to noise variances that increase with the distance from the sensor; this fact considerably affected the depth range of operability of the system in [2], highly decreasing its performance beyond distances further than 7 m away from the camera.

D. Koopman theory

The time evolution of the 3-D skeleton joint coordinates estimated in Sec. II-C during a certain walking activity can also be regarded as an autonomous nonlinear dynamical system,

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k), \quad (4)$$

where $\mathbf{x} \in \mathbb{R}^{21}$ (7 joints with 3 coordinates each), and f is the nonlinear evolution operator. This description enables the exploration of additional techniques for nonlinear systems to predict more accurate joint states. For this purpose, here we introduce a general overview of the discrete-time Koopman operator. We refer the interested reader to [21] for a more in-depth development.

Let \mathbf{x} be an n -dimensional state variable on a manifold $\mathcal{M} \subseteq \mathbb{R}^n$, whose discrete-time evolution is defined by (4), and where $f: \mathcal{M} \mapsto \mathcal{M}$ is the (nonlinear) evolution operator. Moreover, let us define an *observable* function ψ that belongs to the set of functions mapping elements of \mathcal{M} to \mathbb{R} , $\mathcal{F} = \{f: \mathcal{M} \mapsto \mathbb{R}\}$. Then, the Koopman operator \mathcal{K} is defined as an operator acting on ψ such that

$$(\mathcal{K}\psi)(\mathbf{x}) = (\psi \circ f)(\mathbf{x}) = \psi(\mathbf{x}_{k+1}). \quad (5)$$

Equation (5) describes a new dynamical system in \mathcal{F} , where \mathcal{K} determines the evolution dynamics of the observable function ψ , contrary to f which acts directly on \mathcal{M} (\circ is the composition operator). Additionally, this definition of the Koopman operator holds two interesting properties. On the one hand, since \mathcal{F} is infinite dimensional, then also \mathcal{K} must be infinite dimensional, thus rendering this exact definition impractical for real-life applications. On the other hand, \mathcal{F} is a vector-valued space and the composition operator is linear on this space, and hence the Koopman operator is also linear, even if the initial dynamics governing f are nonlinear. The data-driven Extended Dynamic Mode Decomposition (EDMD) [21] method approximates the infinite-dimensional Koopman operator \mathcal{K} by a finite-dimensional operator \mathbf{K} (a matrix), which in turn yields a linear approximate representation of the initial system.

EDMD requires a training dataset D with M consecutive pairs of system states, $D = \{\{\mathbf{x}_i, f(\mathbf{x}_i)\}_{i=0}^{M-1}\}$. Additionally, let us define a finite set of P observable functions that form the column vector-valued function $\Psi: \mathcal{M} \mapsto \mathbb{R}^P$ (which are typically chosen by the designer, see Sec. III-C),

$$\Psi(\mathbf{x}) = [\psi_1(\mathbf{x}), \psi_2(\mathbf{x}), \dots, \psi_P(\mathbf{x})]^\top. \quad (6)$$

By combining equations (5) and (6) we arrive at the following expression, which is linear in Ψ :

$$\Psi(\mathbf{x}_{k+1}) \approx \mathbf{K}\Psi(\mathbf{x}_k). \quad (7)$$

The approximate Koopman matrix \mathbf{K} can be computed from (7) in closed form using all data points in D by solving its corresponding least-squares problem [21].

Remark 1 (Predicting \mathbf{x}_{k+1}): The system state \mathbf{x}_{k+1} can be predicted using the approximate Koopman operator by using the identity map as one of the vector-valued observable functions [22], namely

$$\Psi(\mathbf{x}) = [\mathbf{x}^\top, \psi_1(\mathbf{x}), \psi_2(\mathbf{x}), \dots, \psi_P(\mathbf{x})]^\top. \quad (8)$$

The approximate prediction $\hat{\mathbf{x}}_{k+1}$ is then extracted with

$$\hat{\mathbf{x}}_{k+1} = \mathbf{K}^{[n]}\Psi(\mathbf{x}_k) \approx \mathbf{x}_{k+1}, \quad (9)$$

where $\mathbf{K}^{[n]}$ corresponds to the first n rows of the Koopman matrix \mathbf{K} . Note that the extended $\Psi(\mathbf{x})$ needs to be defined prior to computing \mathbf{K} .

E. Spatial Temporal Graph Convolutional Network

Sec. II-D considers a way to create a model of human movement as an autonomous dynamical system. This model might be reasonable when a single activity is considered (e.g., walking). However, real-life data is likely to comprise a sequence of multiple activities, such as *standing up* \rightarrow *standing* \rightarrow *walking*. In this case, we posit that it is more natural to model the full sequence as the evolution of a hybrid dynamical system with three underlying and fundamentally different models. We therefore propose to use a classifier that can distinguish between activities to select the right model. In addition, the graph-like structure of the human skeleton and the correlations between the motions suggest utilizing an architecture that can capture this organization. For this purpose, we use a Spatial Temporal Graph Convolutional Network [15] architecture (ST-GCN), which we review below.

An ST-GCN is a neural network that receives both the joint coordinates from a sequence of keyframes and information on the graph structure, and performs a series of spatio-temporal (ST) convolutions to extract features both in the spatial domain and the temporal domain. Each ST layer implements one convolution of each type.

Spatial Convolution: This convolution exploits the natural intra-skeleton connections (edges) between the joints on a single keyframe, and is defined as

$$f_{out}(v_i) = \sum_{v_j \in B(v_i)} \frac{1}{Z_{ij}} f_{in}(v_j) w(l_i(v_j)). \quad (10)$$

f_{in} and f_{out} are the input and output features for a given node, respectively, v_i is the center node where the convolution is computed, $B(v_i)$ corresponds to the set of 1-distance *spatial neighbors* of node v_i , and w is the learnable weight variable. While this expression resembles the standard 2-D image convolution, the main difference lies on the partitioning strategy used to divide the neighbors into subsets, governed by the mapping function l_i . In particular, $l_i(v_j)$ maps each

$v_j \in B(v_i)$ to one of 3 different subsets: the center joint v_i , the centripetal group (neighbors closer to the centroid of the skeleton than v_i), or the centrifugal group (neighbors further from the center than v_i). Then, Z_{ij} becomes a scaling factor that accounts for unbalanced neighbor contributions to each group.

The spatial convolution from equation (10) can be implemented in tensor form [23] for each frame t in T as

$$\mathbf{F}_{out,t} = \sum_{j=1}^3 \Lambda_{jt}^{-\frac{1}{2}} (\mathbf{A}_{jt} \odot \mathbf{M}) \Lambda_{jt}^{-\frac{1}{2}} \mathbf{F}_{in,t} \mathbf{W}_j. \quad (11)$$

$\mathbf{F}_{in,t}$ and $\mathbf{F}_{out,t}$ are slices on the temporal dimension of the input and output tensors, of shape $[N, T, C_{in}]$ and $[N, T, C_{out}]$ respectively (N corresponds to the number of joints, T denotes the temporal dimension, C is the number of channels), and \mathbf{W}_j represents the weight tensor for group j , of shape $[C_{in}, C_{out}]$. The structure of the graph connectivity is encoded by the $N \times N$ matrices \mathbf{A}_{jt} , which represent the adjacency matrices of all one-hop neighbors on each subset; Λ_{jt} , which are diagonal matrices whose diagonal elements are $\Lambda_{jt}^{ii} = \sum_{m=0}^{N-1} \mathbf{A}_{jt}^{im} + \alpha$ (with $\alpha \ll 1$ to avoid singularities); and \mathbf{M}_j , which are mask matrices that learn the importance of each edge connection (\odot denotes the element-wise product).

Temporal Convolution: The temporal convolution is applied on the output tensor of the spatial convolution phase. It exploits the inter-frame connections of the same joint between adjacent keyframes by performing a standard 1-D convolution on the dimension T of \mathbf{F}_{out} . Hence, in this case the *temporal neighbors* of node v_{it} are defined as $B(v_{it}) = \{v_{iq} | -\Gamma/2 \leq q - t \leq \Gamma/2\}$, where Γ is the temporal kernel size.

Lastly, the output of the final ST layer is sent to a fully connected (FC) layer, which predicts the activity label.

III. THE KOOPMAN FACTOR

In this paper, our goal is to build a new factor graph factor that, given a set of estimated joint coordinates $\mathbf{x}_k = [\mathbf{n}_k^0, \dots, \mathbf{n}_k^6]^\top$ at time t_k , it 1) identifies its corresponding walking activity a_k , and 2) predicts a *prior* new set of joints at t_{k+1} by selecting a Koopman matrix \mathbf{K}_{a_k} associated to activity a_k . This factor is added to the original factor graph from Sec. II-C, and the schematic of its components and architecture is shown in Fig. 1.

A. Normalization of input variables

The two trainable modules receive the 3-D skeleton joint coordinates as inputs. The map between the system variables $\{\mathcal{R}_k, \mathcal{L}\}$ and a joint \mathbf{n}_k^j is defined as

$$\mathbf{n}_k^j = \mathbf{n}_k^0 + \sum_{(p,q) \in E_j} l^{pq} \mathbf{R}_k^{pq} \mathbf{e}_3, \quad (12)$$

where (p, q) runs across all links E_j in the kinematic chain from the root node \mathbf{n}^0 to the child node \mathbf{n}^j .

To make the system invariant to translation and scaling, the input skeletons to both systems are previously centered around their right foot (to preserve as much skeleton variance

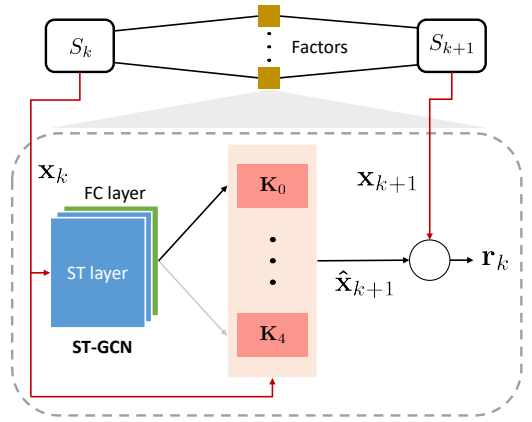


Fig. 1: The schematic of the Koopman factor on a two-state window. The state at S_k is used to compute the joint coordinates \mathbf{x}_k , which are sent to the activity selector. The selected Koopman matrix predicts the next joint states, which in turn are used to compute the residual of the factor.

information as possible [24]) and scaled down to be within the range $[0, 1]$. We denote this centering and normalization operation as \mathcal{N} , and it is applied to each individual joint \mathbf{n}^j :

$$\bar{\mathbf{n}}_k^j = \mathcal{N}(\mathbf{n}_k^j) = \frac{(\mathbf{n}_k^j - \mathbf{n}_k^{(3)}) - c_{min}}{c_{max} - c_{min}}. \quad (13)$$

$\mathbf{n}_k^{(3)}$ are the coordinates of the 3rd joint, which corresponds to the right foot, and are used for centering the skeleton. c_{min} and c_{max} are scalars that refer to the minimum and maximum coordinate values over the whole training dataset. These values are stored in memory for the normalization of other datasets. Note that the vector \mathbf{x}_k can be built by concatenating all seven \mathbf{n}_k^j .

B. ST-GCN for per-frame walking activity selection

Given a temporal sequence of skeleton joint coordinates, we seek to build a classifier that determines the *walking activity type* for each individual keyframe k . For this study we divided the walking activities into five different labels a : *walking*, *standing*, *sitting*, *standing up*, and *sitting down*, such that $a \in [0, \dots, 4]$. These categories fully encompass all the activities performed by the participants across all trajectories studied in Sec. IV.

Many of the GCN-based activity recognition studies in the literature focus on predicting a single activity label from a complete video sequence. In these cases, the temporal dimension usually comprises the entirety of each trajectory. However, our specific application demands *per-frame* labeling. This fact highly constrains the temporal dimension size T , and thus requires splitting each full sequence into multiple independent subset trajectories of a much smaller T size. This restriction also has a direct impact on the selection of both the temporal kernel size Γ and the number of ST layers. Regarding the spatial convolution, the raw data from the joint trajectories is used to pre-compute the tensors \mathbf{A}_j and

Λ_j taking into account the link connections in the skeleton structure.

C. Koopman observable functions

One core element in the construction of the Koopman matrix is the choice of the basis functions to form the vector of observables Ψ . Many different strategies can be followed, which highly depend on the application in hand. However, there is a lack of literature on the application of Koopman theory to human gait estimations due to its novelty. For this reason, we empirically selected the univariate Fourier basis [25] due to its simplicity and good performance, and encourage future research to study the impact the choice of basis functions has on the quality of the gait estimations.

Let $\mathbf{x} \in \mathbb{R}^d$. Its univariate n th-order Fourier basis is defined as the set of functions such that

$$\psi_n(\mathbf{x}) = \cos(\pi \mathbf{c}_n^\top \mathbf{x}), \quad (14)$$

where $\mathbf{c}_n = [c_0, \dots, c_{d-1}]^\top$ is a coefficient vector of size d , and $c_j \in [0, \dots, n]$ for $c_j \in \mathbf{c}$. This implies that for the d -dimensional variable \mathbf{x} , its Fourier basis is composed of $(n+1)^d$ basis functions, a number that suffers from the curse of dimensionality. Since in our application $\mathbf{x} \in \mathbb{R}^{21}$, this representation becomes initially impractical. In this exploratory study, we decouple the three dimensions xyz of the joint coordinates and feed them to \mathbf{K} independently, which allows reducing the feature dimension from 21 to 7. This heuristic approach yielded mean test errors of less than 3 cm per frame. Empirically, for this application we selected $n = 1$, and applying Remark 1 it renders Koopman matrices $\mathbf{K} \in \mathbb{R}^{135 \times 135}$.

D. The Koopman factor

Given two consecutive gait state estimations, $\mathcal{S}_k = \{\mathcal{R}_k, \mathcal{L}\}$ and $\mathcal{S}_{k+1} = \{\mathcal{R}_{k+1}, \mathcal{L}\}$, the Koopman factor performs the following operations sequentially (Fig. 2):

- 1) Obtains the joint coordinates \mathbf{x}_k and \mathbf{x}_{k+1} from the factor graph estimation,
- 2) normalizes \mathbf{x}_k by applying equation (13),
- 3) computes the observable vectors $\Psi_x(\bar{\mathbf{x}}_k)$, $\Psi_y(\bar{\mathbf{x}}_k)$, $\Psi_z(\bar{\mathbf{x}}_k)$ associated to each coordinate of $\bar{\mathbf{x}}_k$,
- 4) predicts activity $a_k(\bar{\mathbf{x}}_k)$ and selects \mathbf{K}_{a_k} ,
- 5) predicts the coordinates of the joint states $\hat{\mathbf{x}}_{k+1}$, and
- 6) computes the residual $\mathbf{r}_k = \hat{\mathbf{x}}_{k+1} - \mathbf{x}_{k+1} \in \mathbb{R}^{21}$.

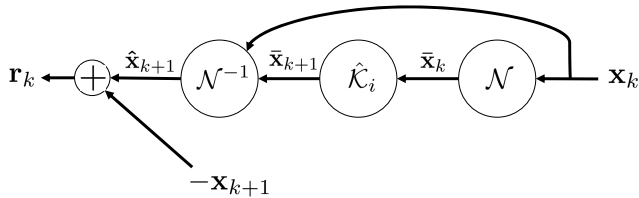


Fig. 2: The computational graph with the operations performed inside the Koopman factor. \mathcal{N} and \mathcal{N}^{-1} are the normalization map and its inverse respectively, and $\hat{\mathcal{K}}_i$ is our Koopman operator applied to coordinate i of $\bar{\mathbf{x}}_k$.

$\hat{\mathcal{K}}_i = \mathbf{K}_{a_k}^{[7]} \Psi_i$ is our Koopman map applied on coordinate $i \in \{x, y, z\}$, \mathcal{N}^{-1} is the inverse of equation (13), and $\bar{\mathbf{x}}_{k+1}$ is obtained by stacking the three coordinate column vectors in the first dimension, $\bar{\mathbf{x}}_{k+1} = [(\bar{\mathbf{x}}_{k+1})_x, (\bar{\mathbf{x}}_{k+1})_y, (\bar{\mathbf{x}}_{k+1})_z]^\top$. Note that to make the centering of the skeleton consistent, the right foot coordinates $\mathbf{n}_k^{(3)}$ are also used in \mathcal{N}^{-1} to recover $\hat{\mathbf{x}}_{k+1}$.

Jacobians. As required by most optimization-on-manifold solvers, explicit expressions for Jacobians of the residual with respect to each variable are required. In the remainder of this section, we provide expressions for the Jacobians of \mathbf{r}_k for coordinate i of an arbitrary node \mathbf{n}_k^j with respect to a parent link (p, q) . We also drop the sub-index k for an easier readability. By applying the chain rule, we can express the Jacobian of \mathbf{r}^j with respect to a general variable \mathbf{X} as

$$\frac{\partial \mathbf{r}_i^j}{\partial \mathbf{X}} = \frac{\partial \mathcal{N}^{-1}}{\partial \hat{\mathcal{K}}_i} \frac{\partial \hat{\mathcal{K}}_i}{\partial \mathcal{N}} \frac{\partial \mathcal{N}}{\partial \mathbf{X}} + \frac{\partial \mathbf{n}_i^{(3)}}{\partial \mathbf{X}} - \frac{\partial (\mathbf{x}_{k+1})_i}{\partial \mathbf{X}}. \quad (15)$$

We provide the mathematical expressions for each term in (15) below:

$$\frac{\partial \mathcal{N}^{-1}}{\partial \hat{\mathcal{K}}_i} = (c_{max} - c_{min}), \quad (16a)$$

$$\frac{\partial \hat{\mathcal{K}}_i}{\partial \mathcal{N}} = \mathbf{K}_{a_k}^{[7]} \left[\mathbf{I}_7, [-\sin(\pi \mathbf{c}_m^\top \bar{\mathbf{x}}_i) \pi \mathbf{c}_m]_{\mathbf{c}_m \in \mathcal{C}} \right]^\top, \quad (16b)$$

$$\frac{\partial \mathcal{N}}{\partial \mathbf{X}} = \frac{1}{c_{max} - c_{min}} \left(\frac{\partial \mathbf{x}_i}{\partial \mathbf{X}} - \frac{\partial \mathbf{n}_i^{(3)}}{\partial \mathbf{X}} \right). \quad (16c)$$

\mathbf{x}_i is a 7-D vector containing coordinate i for all joints, \mathbf{I}_7 corresponds to the 7-D identity matrix, and \mathcal{C} corresponds to the set of all frequencies \mathbf{c}_m included in Ψ_i . The task to compute the Jacobians with respect to each state variable is now reduced to providing individual expressions for the Jacobians of each joint coordinate, analyzed below.

1) *Jacobian with respect to \mathcal{R}_k :* Recall that the set of states \mathcal{R}_k is defined as the concatenation of all \mathbf{R}_k^{ij} for $(i, j) \in E$ and \mathbf{n}^0 . The derivatives of a joint \mathbf{n}^j with respect to a parent \mathbf{R}^{pq} and the root joint \mathbf{n}^0 can be expressed as

$$\frac{\partial \mathbf{n}^j}{\partial \mathbf{R}^{pq}} = [-\mathbf{R}^{pq} \mathbf{e}_2, \mathbf{R}^{pq} \mathbf{e}_1, \mathbf{0}_{3 \times 1}]^{l^{pq}}, \quad (17a)$$

$$\frac{\partial \mathbf{n}^j}{\partial \mathbf{n}^0} = \mathbf{I}_3. \quad (17b)$$

2) *Jacobian with respect to \mathcal{L} :* \mathcal{L} refers to the concatenation of all link lengths l^{ij} for $(i, j) \in E$. The derivative of \mathbf{n}^j with respect to a parent l^{pq} is

$$\frac{\partial \mathbf{n}^j}{\partial l^{pq}} = \mathbf{R}^{pq} \mathbf{e}_3. \quad (18)$$

Equations (17) and (18) should be computed for each joint and rearranged to form three 7-D column vectors, corresponding to each coordinate x, y and z , before plugging them into (15).

IV. EXPERIMENTAL STUDY

A. Dataset

The dataset used in this study comprises 65 trajectories across 5 participants corresponding to four different standardized lower limb mobility tests, namely the 10 Meter Walk Test (10MWT), the Functional Gait Assessment (FGA), the Short Physical Performance Battery test (SPPB), and the Timed Up and Go test (TUG). Each trajectory contains ground truth data from the Qualisys motion capture system at 200 Hz, RGB and point cloud data at 30 Hz from an Intel RealSense D435 [26], and IMU data at 120 Hz. This dataset was recorded under the IRB protocol Mass General Brigham 2020P003474, and the data was managed by the Robotic Operating System (ROS) [27] middleware.

To build a joint trajectory dataset from the recorded data, we first applied the original system in [2] to each individual recording, which resulted in 65 different joint trajectories, each one comprising approximately between 150 to 600 keyframes (depending on the activity performed). These trajectories contained skeleton keyframes up to 7 m away from the camera, to remain in the operability range of the system. To make the systems rotation invariance, each skeleton keyframe belonging to the training dataset was duplicated at different rotation angles around the vertical axis after the centering operation.

B. ST-GCN training

Considering the requirements addressed in Sec. III-B, our ST-GCN is composed of two spatio-temporal layers with a temporal kernel size of $\Gamma = 7$, with 128 and 256 output channels respectively, and a fully connected layer. This architecture sets the input tensor to be of shape $[B, C_{in}, N, T]$ with $C_{in} = 3$ spatial coordinates (x , y and z), $N = 7$ joints, and $T = 13$ keyframes. B corresponds to the batch dimension. After centering each skeleton, we duplicated each keyframe 7 times while rotating the skeleton at intervals of 45° around the vertical axis. Then we split it into train and test stratified sets, resulting in 66645 and 28563 samples per set (70% and 30% of the full dataset respectively). Each class on the train set has 13109, 4609, 3472, 3041 and 42414 samples. We implemented our ST-GCN using the Pytorch library [28].

To validate the choice of selecting joint coordinates as input features to our network, and since our datasets already contain IMU information (unlike all the other cited activity recognition references), we compared our joint-based ST-GCN activity selector with an IMU-only ST-GCN baseline with a similar architecture. This baseline network takes the pre-integrated IMU velocity measurements as inputs, and the spatial subsets were generated by comparing the root node and its neighbors to the average speed of the four IMUs.

The joint-based ST-GCN output 99.97% test accuracy, while the IMU-only baseline stayed at 92%. Additionally, many of its misclassification errors happened between the standing and sitting classes. This makes sense intuitively, as in both cases the IMU angular velocities are very close to zero. We leave for future work more thorough studies on the use of IMU-only data for activity recognition.

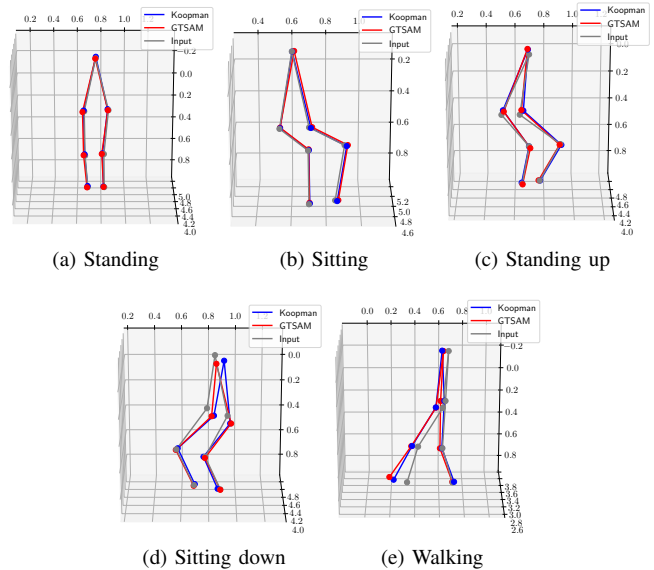


Fig. 3: Comparison between the Koopman predictions with their corresponding GTSAM estimations for one test sample of each one of the five selected walking activities. In gray, the GTSAM skeleton at t_k ; in red, the GTSAM estimation at t_{k+1} ; in blue, the Koopman prediction at t_{k+1} .

C. Koopman training

To construct the Koopman training dataset, all joint trajectories were segregated in pairs of consecutive keyframes $\{\mathbf{x}_k, \mathbf{y}_k\}$. Contrary to the dataset used to train the ST-GCN, both keyframes were centered around the same right foot coordinates of \mathbf{x}_k to match the computational graph from Fig. 2. Then, both skeletons of each pair were duplicated 23 times and rotated at intervals of 15° around their vertical axes. Finally, the pairs were assigned an activity label a_k corresponding to \mathbf{x}_k . This allows training one Koopman matrix \mathbf{K}_{a_k} for each activity independently but following the same procedure. The five Koopman matrices were trained following equation (7) and stored in memory. Fig. 3 shows five frame comparisons belonging to the test sets between the Koopman predictions and their corresponding GTSAM estimations.

D. Experimental analysis

We analyzed the performance of our newly proposed factor by comparing its results with a naive vision-only approach (consisting of image and depth data), with the GTSAM baseline system from [2], and with their corresponding ground truth trajectories.

To study the similarities between the estimated skeleton and its ground truth, we first centered the skeletons from each keyframe of every source around their centroid; then, we computed the Euclidean distance error with respect to the ground truth for all joints across all keyframes in the dataset. This approach allows removing the trajectory estimation bias while focusing solely on the skeleton form. Fig. 4 shows a boxplot comparison of all the Euclidean errors across all keyframes of all trajectories between the vision-only approach,

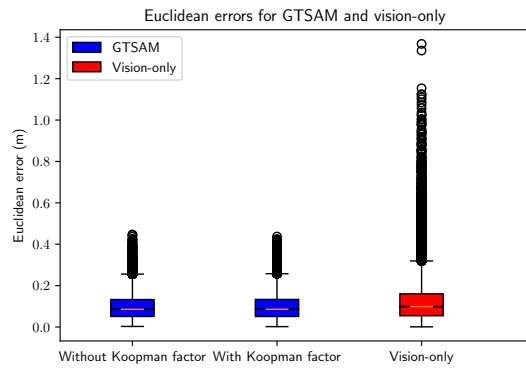


Fig. 4: Boxplot comparison of the Euclidean errors of all joints across all trajectories with respect to their ground truth counterparts.

the baseline system, and the modified factor graph. While both factor graphs highly improve the naive vision-only approach (they reduce the outliers by almost 1 m), there is no considerable difference between the quality of the skeletons from both systems. This was to be expected, since the Koopman matrices were trained purely with the baseline estimations.

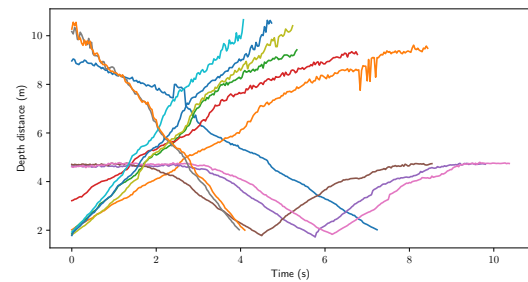
However, the main contribution of the Koopman factor lies on the trajectory estimations at far distances. Figs. 5a and 5b show the evolution of the depth coordinate of the skeleton centroid for a few selected trajectories across multiple tests and participants. The vision-only trajectories start getting distorted around 3 m, which is the range limit recommended by the manufacturer. While the baseline system was able to highly improve the skeleton estimated forms, it still struggles to track the walking kinematics of the skeleton. Fig. 5c shows the same trajectories estimated by the new system with the Koopman factor. Its results highly contrast with the previous two approaches, with much smoother trajectories and no sudden jumps on the estimated depths. Paired with the results from Fig. 4, they suggest that the new estimated skeleton trajectories preserve much better the nature of human walking, which largely amplifies the range of operation of the system to more than 10 m.

A more visual representation of this behavior can be seen in Fig. 6, which compares the results of the system with and without the Koopman factor on three consecutive keyframes. While the depth joint coordinates of the original system's estimations fluctuate in an unnatural manner, the Koopman factor allows preserving the natural flow of the human walking away from the camera.

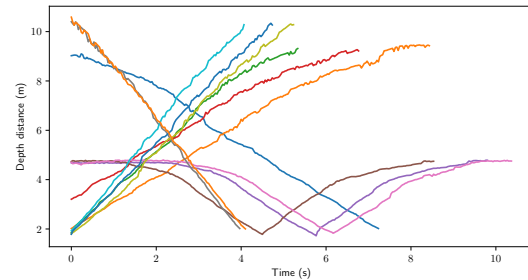
V. CONCLUSIONS AND FUTURE WORK

In this paper, we studied the viability of Koopman theory applied to the field of human pose recognition, targeted at the application of clinical assessment of lower limb mobility.

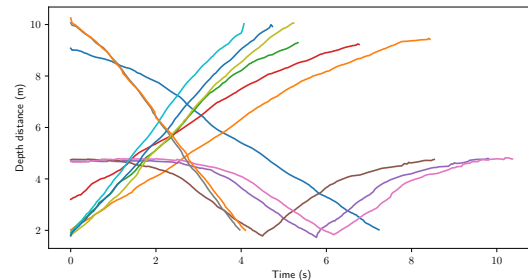
The novelty of our approach lies on the use of Koopman predictions of consecutive human skeleton keyframes within the optimization loop of a factor graph problem to influence the optimized skeleton estimations toward temporal consistency. More specifically, we implemented a new factor graph



(a) Vision-only trajectories



(b) Without the Koopman factor



(c) With the Koopman factor

Fig. 5: Comparison of some sample trajectories of the skeleton centroid estimated by (a) a vision-only approach, (b) the baseline factor graph system, and (c) the modified factor graph with the Koopman factor.

factor that acts in parallel to the main IMU, image, depth, and contact factors of the original factor graph. This new factor is applied on each individual keyframe of the trajectory, and runs a two-step process. First, a trained spatial temporal graph convolutional network detects the walking activity the participant is performing among the selected set of *standing*, *sitting*, *standing up*, *sitting down*, and *walking*. Then, this prediction is used to select a Koopman operator trained on the specified activity, and computes a measurement-free *a priori* skeleton estimation of the skeleton in the following keyframe. This estimation is based on a data-driven learnt representation of the inherent dynamics of each one of the five selected walking activities, and thus complements the information provided by the external measurements. Additionally, we validated the selection of the joint coordinates as features for the activity recognition system over an IMU-only alternative.

We studied the performance of the proposed Koopman factor by running our system against custom datasets composed of standard clinical tests for lower limb mobility

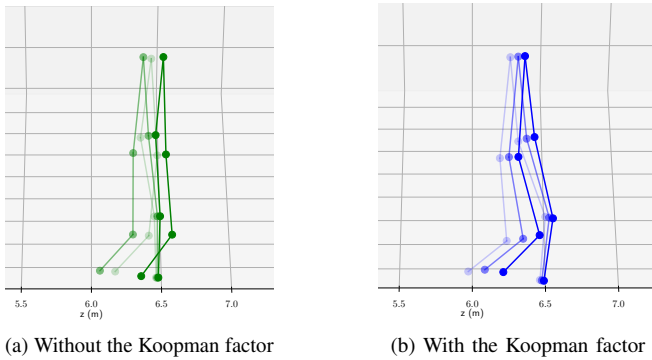


Fig. 6: Three consecutive keyframes (with increasing opacity) corresponding to the side view of a skeleton walking. On the left, the estimations obtained using the original factor graph system. On the right, the same keyframes estimated with the new Koopman factor.

assessment. Not only was the modified factor graph able to preserve the quality of the estimated skeletons observed in the original system, but it also showed a substantial improvement in filtering out the noise introduced by the depth sensor, extending the depth range of operation of the system to more than 10 m away from the camera.

In future work, we plan to study more rigorous alternatives on the Koopman observables and frequencies selection, which should enable even better predictions without having to increase the Koopman dimensionality. Additionally, we plan to analyze the effect of the Koopman factor in those images where the human view is partially or completely blocked. Finally, embedding the human walking dynamics into a Koopman representation could allow reducing the number of required IMUs. This fact would be of major importance toward building a clinical user-friendly system for automatic mobility assessment in the home.

REFERENCES

- [1] Centers for Medicare and Medicaid Services, “National health expenditure fact sheet,” Available at: <https://www.cms.gov/Research-Statistics-Data-and-Systems/Statistics-Trends-and-Reports/NationalHealthExpendData/NHE-Fact-Sheet.html>, 2013, accessed: 2021-03-20.
- [2] M. Mitjans, M. Theofanidis, A. N. Collimore, M. L. Disney, D. M. Levine, L. N. Awad, and R. Tron, “Visual-inertial filtering for human walking quantification,” in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 13 510–13 516.
- [3] L. Bai, M. G. Pepper, Y. Yan, S. K. Spurgeon, M. Sakel, and M. Phillips, “Quantitative assessment of upper limb motion in neurorehabilitation utilizing inertial sensors,” *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 23, no. 2, pp. 232–243, 2014.
- [4] Y. Tian, X. Meng, D. Tao, D. Liu, and C. Feng, “Upper limb motion tracking with the integration of imu and Kinect,” *Neurocomputing*, vol. 159, pp. 207–218, 2015.
- [5] T. Von Marcard, G. Pons-Moll, and B. Rosenhahn, “Human pose estimation from video and imus,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 8, pp. 1533–1547, 2016.
- [6] A. K. Patil, A. Balasubramanyam, J. Y. Ryu, B. Chakravarthi, and Y. H. Chai, “An open-source platform for human pose estimation and tracking using a heterogeneous multi-sensor system,” *Sensors*, vol. 21, no. 7, p. 2340, 2021.
- [7] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, “OpenPose: realtime multi-person 2D pose estimation using Part Affinity Fields,” *arXiv preprint arXiv:1812.08008*, 2018.
- [8] Y. Raaj, H. Idrees, G. Hidalgo, and Y. Sheikh, “Efficient online multi-person 2D pose tracking with recurrent spatio-temporal affinity fields,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4620–4628.
- [9] M. Kocabas, S. Karagoz, and E. Akbas, “Self-supervised learning of 3d human pose using multi-view geometry,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1077–1086.
- [10] N. Nakano, T. Sakura, K. Ueda, L. Omura, A. Kimura, Y. Iino, S. Fukushima, and S. Yoshioka, “Evaluation of 3d markerless motion capture accuracy using openpose with multiple video cameras,” *Frontiers in sports and active living*, vol. 2, p. 50, 2020.
- [11] X. Zhou, Q. Huang, X. Sun, X. Xue, and Y. Wei, “Towards 3D human pose estimation in the wild: a weakly-supervised approach,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 398–407.
- [12] S. Zhang, Y. Wang, and A. Li, “Cross-view gait recognition with deep universal linear embeddings,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 9095–9104.
- [13] N. Heidari and A. Iosifidis, “Temporal attention-augmented graph convolutional network for efficient skeleton-based human action recognition,” in *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE, 2021, pp. 7907–7914.
- [14] X. Zhang, C. Xu, X. Tian, and D. Tao, “Graph edge convolutional neural networks for skeleton-based action recognition,” *IEEE transactions on neural networks and learning systems*, vol. 31, no. 8, pp. 3047–3060, 2019.
- [15] S. Yan, Y. Xiong, and D. Lin, “Spatial temporal graph convolutional networks for skeleton-based action recognition,” in *Thirty-second AAAI conference on artificial intelligence*, 2018.
- [16] F. R. Kschischang, B. J. Frey, and H. A. Loeliger, “Factor graphs and the sum-product algorithm,” *IEEE Transactions on information theory*, vol. 47, no. 2, pp. 498–519, 2001.
- [17] H. A. Loeliger, “An introduction to factor graphs,” *IEEE Signal Processing Magazine*, vol. 21, no. 1, pp. 28–41, 2004.
- [18] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, “IMU preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation.” Georgia Institute of Technology, 2015.
- [19] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.
- [20] F. Dellaert, “Factor graphs and gtsam: A hands-on introduction,” Georgia Institute of Technology, Tech. Rep., 2012.
- [21] M. O. Williams, I. G. Kevrekidis, and C. W. Rowley, “A data-driven approximation of the koopman operator: Extending dynamic mode decomposition,” *Journal of Nonlinear Science*, vol. 25, no. 6, pp. 1307–1346, 2015.
- [22] I. Abraham, G. De La Torre, and T. D. Murphey, “Model-based control using koopman operators,” *arXiv preprint arXiv:1709.01568*, 2017.
- [23] T. N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” *arXiv preprint arXiv:1609.02907*, 2016.
- [24] J. P. Vox and F. Wallhoff, “Preprocessing and normalization of 3d-skeleton-data for human motion recognition,” in *2018 IEEE Life Sciences Conference (LSC)*. IEEE, 2018, pp. 279–282.
- [25] G. Konidaris, S. Osentoski, and P. Thomas, “Value function approximation in reinforcement learning using the fourier basis,” in *Twenty-fifth AAAI conference on artificial intelligence*, 2011.
- [26] Intel, “Depth camera d435/,” Available at: <https://www.intelrealsense.com/depth-camera-d435/>, 2021, accessed: 2021-11-10.
- [27] Stanford Artificial Intelligence Laboratory et al., “Robotic operating system,” Available at: <https://www.ros.org>, 2018.
- [28] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, “Pytorch: An imperative style, high-performance deep learning library,” in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035.