書き順と筆触を独立に学ぶ階層型CVAE

○會田真広 (埼玉大学)　菅原慶人 (埼玉大学)
境野翔 (筑波大学)　辻俊明 (埼玉大学)

1. 序論

現在, 人の作業の多くはロボットへの代替が進んでいる. しかし, 接触を伴うなどの複雑な動作は, プログラムで 1 から作り込むことが非常に困難である. プログラミングのコストを削減するために, 人間の動作を利用し, 教示する模倣学習が研究されている [1].

取得した人の動作データを用いて, 生成モデルを構築する手法の 1 つとして変分オートエンコーダ (Variational AutoEncoder:VAE)[2] が挙げられる. VAE は, Encoder と Decoder の 2 つのモデルから構成されており, 入力データと低次元の潜在変数の対応関係を学習するニューラルネットワークである. VAE を拡張した手法として条件付き変分オートエンコーダ (Conditional VAE:CVAE)[3] が挙げられる. CVAE は Encoder, Decoder に入力と共にラベルを付与し, 生成時に Decoder に対して所望のラベルを入れることで, 望んだ出力を得られるため, 画像 [4] や音楽 [5] の生成, 異常検知 [6] に活用されている.

Ha らは VAE で構成された sketch-rnn(recurrent neural network) を用いて, スケッチ画像を時系列ベクトルで構成された画像としてとらえることにより, 書き順を保持した画像の生成をした [7]. Uegaki らは人間が教示した書字動作の時系列データベースから，VAE によって動作生成モデルを構築し，動作の種類を表すラベルを用いることで所望の種類の動作を生成する手法を述べた [8]. また, Zhang らは RNN を用いて漢字を書く際のペンの動作をモデル化することにより, ベクトル化された漢字の生成を試みた [9]. しかし, これら研究では, 複数の動作を 1 つのモデルで生成できるが, 新たな動作を生成するためには, 人が新しく動作を取得し, 学習し直す必要が発生する, あるいは事前に大量の学習データを必要とすることが考えられる.

高次の動作を学習する為には階層型のモデルが有効とされており [10][11], 学習を階層化することで, 必要とする学習データの量を軽減できると示されている [12].

そこで, 本研究では, 二つの動作を異なるモデルで独立に学習し, 学習済みモデルを組み合わせて軌道を生成することで, 二つの学習した動作の特徴を有した軌道を生成する手法を提案する. 人が文字・模様を書く際の動作を独立に学習し, 学習済みの Decoder を階層型の動作生成モデルとする. モデルを階層型に組み合わせることで学習時には存在しない, 学習した動作の特徴を有した軌道の生成が出来ると考えられる.

以下に本論文の構成を示す. まず, 2 節で提案手法を述べ, 3 節で実験の内容とその結果により提案手法の有効性を示す. 最後に 4 節でまとめを記述する.

2. 提案手法

本節では, 提案手法である階層型 CVAE の構造とモデルの学習方法について述べる. また, 本研究では人が文字・模様を書く際の軌道を学習データとして利用しており, 文字を書く際の順序を書き順, 特徴的な軌道を筆触と表現する.

2.1 階層型 CVAE の構造・学習

本研究では, ①書き順を学習し, 軌道の始点, 終点を生成するモデルを $\mathrm{VAE_{point}}$, ②筆触を学習し, 軌道を生成するモデルを $\mathrm{CVAE_{traj}}$ として, それぞれを独立して学習する.

①で取得した動作を $\boldsymbol{x_m^{\mathbf{point}}}$ と表記し, 取得した動作から抽出した始点, 終点をそれぞれ $\boldsymbol{x_m^{\mathbf{start}}}, \boldsymbol{x_m^{\mathbf{end}}}$ で表す. ただし, 添え字 $\boldsymbol{m}$ は軌道の本数番号を表す. また, 始点・終点の両者まとめたものを $\boldsymbol{x_m^{\mathbf{p}}}$ とし, 以下の式 (1) で表す.

$$\boldsymbol{x_m^{\mathbf{p}}} = \left(\boldsymbol{x_m^{\mathbf{start}}}, \boldsymbol{x_m^{\mathbf{end}}}\right) \tag{1}$$

$\mathrm{VAE_{point}}$ の構造を図 1 に示す. $\mathrm{VAE_{point}}$ では始点と終点の組, つまり書き順を時系列データとして学習する. ただし, M は軌道の本数を示す. $\mathrm{VAE_{point}}$ を学習する際の損失関数 L_{point} は式 (2) のように表される.

$$L_{\mathrm{point}} = \frac{1}{M}\sum_{i=1}^{M}\left(\hat{\boldsymbol{x}}_{\boldsymbol{i}}^{\mathbf{p}} - \boldsymbol{x}_{\boldsymbol{i}}^{\mathbf{p}}\right)^2 - L_{\mathrm{KL}} \tag{2}$$

ここで, 右辺第一項は VAE の入出力に対する平均二乗誤差, L_{KL} は KL ダイバージェンスと呼ばれる値であり, これは 2 つの確率分布の類似度を表す尺度である. 潜在変数と正規分布の類似度を表す場合, 潜在変数の次元を J とすると KL ダイバージェンスは式 3 で表せる. ただし, σ, μ は Encoder の出力である.

$$L_{\mathrm{KL}} = \frac{1}{2}\sum_{j=1}^{J}\left(1 + \ln(\sigma^2) - \mu^2 - \sigma^2\right) \tag{3}$$

②で取得した動作を $\boldsymbol{x_n^{\mathbf{traj}}}$ と表記する. ただし, 添え字 $\boldsymbol{n}$ はサンプル数を表す. ここで, $\boldsymbol{x_n^{\mathbf{traj}}}$ の全体を軌道の始点でオフセット処理する. $\boldsymbol{x_n^{\mathbf{traj'}}}$ を処理後の軌道とし, 式 (4) で表す.

$$\boldsymbol{x_n^{\mathbf{traj'}}} = \boldsymbol{x_n^{\mathbf{traj}}} - \boldsymbol{x_1^{\mathbf{traj}}} \tag{4}$$

$\mathrm{CVAE_{traj}}$ の構造を図 2 に示す. $\mathrm{CVAE_{traj}}$ では始点と終点が与えられた後に, その二点をつなぐ軌道をモデルに基づき生成する. モデルが直線軌道で学習されたものであれば直線を生成し, 短いタッチの描画を学習したものであれば, 短いタッチの描画軌道を生成する. す

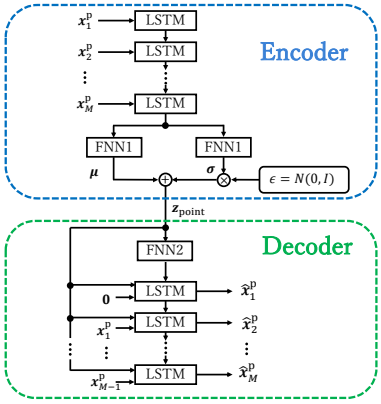

図 1: Construction of $\mathrm{VAE}_{\mathrm{point}}$

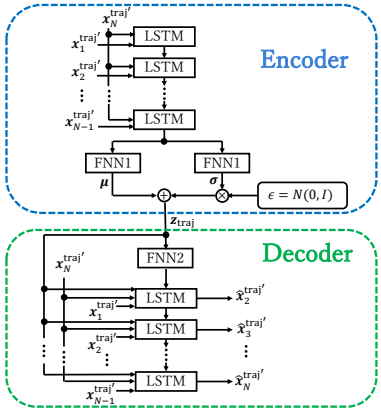

図 2: Construction of $\mathrm{CVAE}_{\mathrm{traj}}$

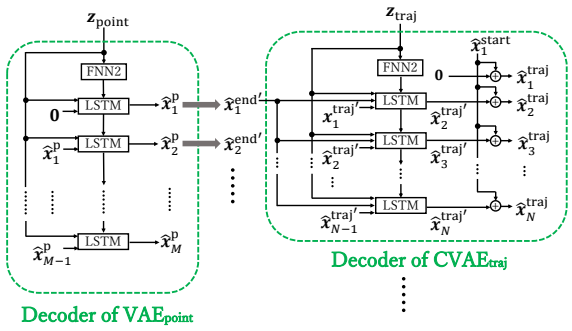

図 3: Generation phase

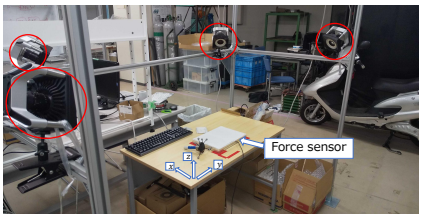

図 4: Motion capture and Force sensor

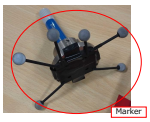

図 5: Marker

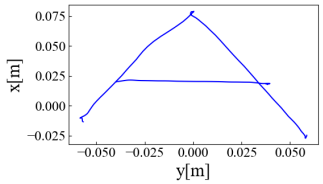

図 6: Character:A

なわち, 筆触に相当する部分を学習するモデルである. 学習時に条件付けをするために, Encoder, Decoder に $\boldsymbol{x}_N^{\mathbf{traj}'}$ を常に加える. ただし, N は軌道のサンプル数を示す. $\mathrm{VAE}_{\mathrm{traj}}$ を学習する際の損失関数 L_{traj} は式 (5) のように表される. ここで, 右辺第一項は CVAE の入出力に対する平均二乗誤差, 第二項は CVAE の入出力の微分値に対する平均二乗誤差, L_{KL} は KL ダイバージェンスである.

$$L_{\mathrm{traj}} = \frac{1}{N}\sum_{i=1}^{N}\left(\hat{\boldsymbol{x}}_i^{\mathbf{traj}'} - \boldsymbol{x}_i^{\mathbf{traj}'}\right)^2 + \frac{1}{N}\sum_{i=1}^{N}\left(\hat{\boldsymbol{x}'}_i^{\mathbf{traj}'} - \boldsymbol{x'}_i^{\mathbf{traj}'}\right)^2 + L_{\mathrm{KL}} \quad (5)$$

2.2 階層型 CVAE による軌道生成

VAE はモデルの学習後, Encoder と Decoder を切り離し, Decoder に潜在変数を与えることで軌道を生成することができる. したがって, $\mathrm{VAE}_{\mathrm{point}}$ と $\mathrm{CVAE}_{\mathrm{traj}}$ の学習終了後, 二つのモデルの Decoder を組み合わせて軌道を生成する. 生成時における構造を図 3 に示す.

まず, $\mathrm{VAE}_{\mathrm{point}}$ の Decoder に対して潜在変数 $\boldsymbol{z_{point}}$ を与え, 軌道の始点 $\hat{\boldsymbol{x}}^{\mathbf{start}}$・終点 $\hat{\boldsymbol{x}}^{\mathbf{end}}$ を出力する. 続いて出力された始点でオフセット処理を施した終点 $\hat{\boldsymbol{x}}^{\mathbf{end}'} = \hat{\boldsymbol{x}}^{\mathbf{end}} - \hat{\boldsymbol{x}}^{\mathbf{start}}$ を求める. この値を $\mathrm{CVAE}_{\mathrm{traj}}$ の終点と対応させる. 得られた $\hat{\boldsymbol{x}}^{\mathbf{end}'}$ と潜在変数 $\boldsymbol{z_{traj}}$ を $\mathrm{CVAE}_{\mathrm{traj}}$ の Decoder に対して入力し, 出力された軌道にオフセット時に引いた始点 $\hat{\boldsymbol{x}}^{\mathbf{start}}$ を足し合わせることで, 二つのモデルを組み合わせた軌道が生成される.

3. 実験

本節では提案手法の有効性の検証のために行った実験内容について述べる.

3.1 実験装置

図 4 に人の動作時の力・位置情報取得に利用した装置を示す. 本実験では, 6 軸力覚センサを埋め込んだ土台型のセンサ [13] を用いた. 本センサを用いることで, 道具にセンサを埋め込むことなく力情報を取得可能である. また, 人の動作で重要となる位置情報は, モーションキャプチャとして高い性能を持つ OptiTrack[14] を利用した. 図 4 中の赤丸で囲われたカメラ 4 台で図 5 のマーカーの位置情報を取得することで人の動作を記録することが出来る. また, 力センサと OptiTrack の記録周期は 8 msec である.

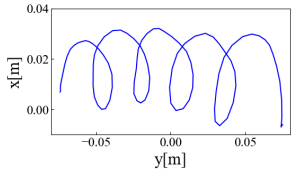

図 7: Trajectory:long

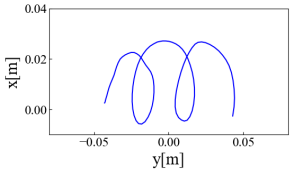

図 8: Trajectory:short

表 1: Parameters of VAEs

	$\mathrm{VAE_{point}}$	$\mathrm{CVAE_{traj}}$
LSTM layer size	1	2
LSTM unit	64	256
Dimension of z_{point}/z_{traj}	6	3
FNN2 activation function	tanh	tanh
Optimizer	Adam	Adam
Batch size	4	10
Epoch	10000	50000

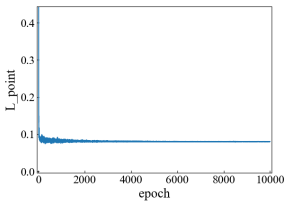

図 9: Loss:$\mathrm{VAE_{point}}$

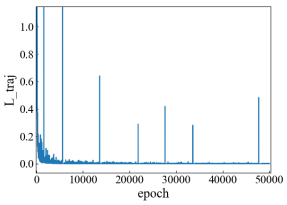

図 10: Loss:$\mathrm{CVAE_{traj}}$

3.2 動作データの収集

本研究では動作データとして, ペンを用いて① 図 6 の軌道を 20 回, ② 図 7 に示す軌道, 図 8 に示す短めの軌道を描く動作をそれぞれ 10 回ずつ取得した. ① で取得したデータでは, (a)「ペンが紙と接触し, 線を書いている動作」と (b)「線の終了地点から次の線の開始地点までの移動」が含まれている. そこで, 文字書き順を取得するため, 接触力が 0.25 N 以上の部分を (a) の範囲とすることで, (b) の範囲を除去し, 文字の始点と終点を抽出した. また, 図 6 は, 3 本の直線で構成されているため, 軌道の本数は 3 本であり, $M = 3, m = 1, 2, 3$ となる. また, ② で取得したデータそのままでは, 学習コストが問題となる可能性がある. そのため, 取得したデータをダウンサンプリングすることにで 100 サンプルにそろえた. したがって $N = 100, n = 1, 2, \cdots, 100$ となる. また, 単一方向のデータのみでは上手く学習できない可能性がある. そこで, 取得した計 10 本のデータを 60° ずつ回転させ, 計 60 本に水増しした.

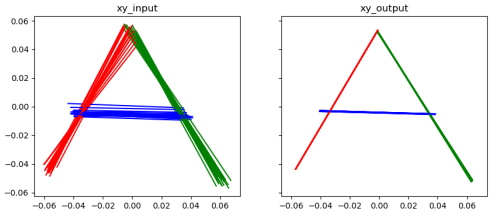

図 11: $\mathrm{VAE_{point}}$ input/output

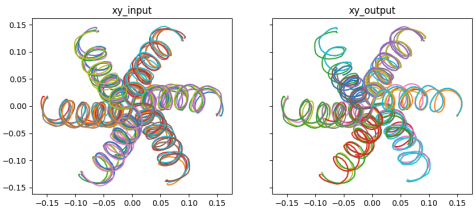

図 12: $\mathrm{CVAE_{traj}}$ input/output

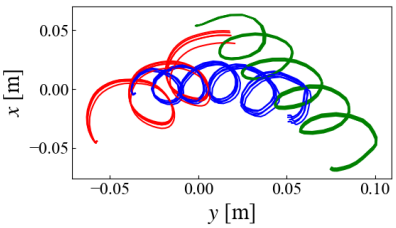

図 13: Combination output

3.3 結果

本研究で使用した学習モデルのパラメータを表 1 に示す. まず始めに, 二つのモデルの学習時に置ける損失の変化を図 9, 図 10 に示す. 続いて, 二つのモデルがそれぞれ独立に生成をした際の結果を図 11, 図 12 に示す. $\mathrm{VAE_{point}}$ での入出力は始点・終点であるが, その二点を繋いでいるものが図 11 のようになっている. この図から書き順通りに 3 セットの始点・終点が出力されていることが分かる. また, $\mathrm{CVAE_{traj}}$ から, 長い軌道と短い軌道がそれぞれ出力されていることが分かる.

次に, 独立して学習した二つのモデルを組み合わせて生成した結果を図 13 に示す. この結果から, $\mathrm{CVAE_{traj}}$ で学習した筆触が $\mathrm{VAE_{point}}$ で出力された始点・終点に対応して生成されていることが分かる.

3.4 $\mathrm{VAE_{point}}$ の入力変更

$\mathrm{VAE_{point}}$ の入力データとして, 図 14 に示す文字を使用し, $\mathrm{CVAE_{traj}}$ のモデルは変更せずに軌道を生成した. これにより二つのモデルを組み合わせて生成した結果を図 15 に示す. ”A” を出力した際と同様に図 14

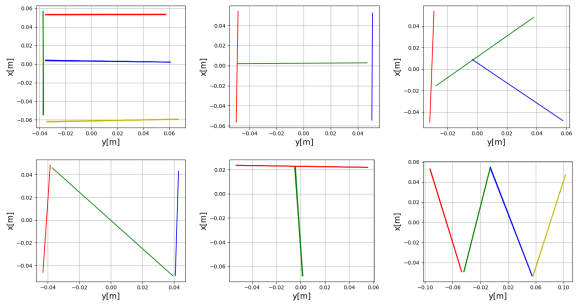

図 14: $\mathrm{VAE}_{\mathrm{point}}$ output (E, H, K, N, T, W)

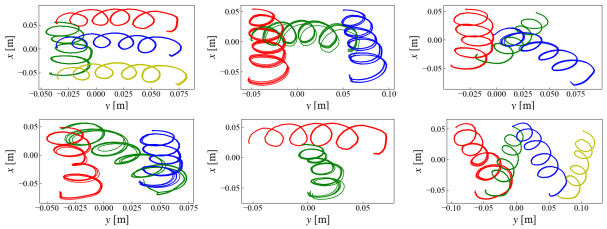

図 15: Combination output (E, H, K, N, T, W)

に示した文字の場合でも, $\mathrm{CVAE}_{\mathrm{traj}}$ で学習した筆触が $\mathrm{VAE}_{\mathrm{point}}$ で出力された始点・終点に対応して生成されていることが分かる. さらに, $\mathrm{CVAE}_{\mathrm{traj}}$ で学習した筆触では縦の軌道は学習していないが, 図 15 では, それぞれ縦線の部分も他と同様に $\mathrm{VAE}_{\mathrm{point}}$ で出力された始点・終点に対応した筆触が再現されていることが分かる.

4. まとめ

本稿では, 二つの動作を異なるモデルで独立に学習を行い, 学習したモデルを組み合わせて軌道を生成することで, 二つの学習した動作の特徴を有した軌道を生成する手法を提案した. 人が文字・模様を書く際の動作のうち, 「書き順を学習し, 軌道の始点・終点を生成するモデル」と, 「筆触を学習し, 軌道を生成するモデル」のそれぞれを独立した VAE で学習する. その後, 学習済みの Decoder を階層型の動作生成モデルとし, 軌道生成時に二つの独立したモデルを階層型に組み合わせることで, 学習時には存在しないが, 学習した動作の特徴を有した軌道を生成した.

この結果から, 学習した筆触が出力された始点・終点に対応して生成されていることが示された. さらに, 学習した筆触モデルでは学習していない方向に出力された, 始点・終点にも同様に学習した筆触が同様に再現されていることが確認された.

参 考 文 献

[1] Schaal, Stefan: “Learning from demonstration,” Advances in neural information processing systems, 1997.

[2] Kingma, Diederik P., and Max Welling: “Auto-encoding variational bayes”, arXiv preprint arXiv:1312.6114, 2013.

[3] Kingma, D. P., Mohamed, S., Rezende, D. J. and Welling, M.: “Semi-supervised learning with deep generative models”, Advances in neural information processing systems, 2014.

[4] N. Nakagawa, R. Togo, T. Ogawa and M. Haseyama: “Face Synthesis via User Manipulation of Disentangled Latent Representation”, 2020 IEEE 9th Global Conference on Consumer Electronics (GCCE), pp. 692-693, 2020.

[5] J. Jiang, G. G. Xia, D. B. Carlton, C. N. Anderson and R. H. Miyakawa: “Transformer VAE: A Hierarchical Model for Structure-Aware and Interpretable Music Representation Learning”, 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 516-520, 2020.

[6] A. A. Pol, V. Berger, C. Germain, G. Cerminara and M. Pierini: “Anomaly Detection with Conditional Variational Autoencoders”, 2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA), pp. 1651-1657, 2019.

[7] Ha, David, and Douglas Eck: “A neural representation of sketch drawings”, arXiv preprint arXiv:1704.03477, 2017.

[8] 上柿 雅裕, 桂 誠一郎: “書字動作の時系列データベースに基づく軌道と接触力の生成”, ロボティクス・メカトロニクス講演会講演概要集, 2020 巻, 1P1-H09, 2020.

[9] Zhang, X. Y., Yin, F., Zhang, Y. M., Liu, C. L. and Bengio, Y.: “Drawing and recognizing chinese characters with recurrent neural network”, IEEE transactions on pattern analysis and machine intelligence 40.4, 849-862, 2017.

[10] Kulkarni, T. D., Narasimhan, K., Saeedi, A. and Tenenbaum, J.: “Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation”, Advances in neural information processing systems 29, 3675-3683, 2016.

[11] 菅原 慶人, 會田 真広, 辻 俊明: “変分オートエンコーダによる人の動作を基にした研磨動作生成”, ロボティクス・メカトロニクス講演会講演概要集, 1A1-E17, 2021.

[12] Le, H., Jiang, N., Agarwal, A., Dudík, M., Yue, Y. and Daumé, H.: “Hierarchical imitation and reinforcement learning”, International conference on machine learning. PMLR, 2018.

[13] N. Totsu, S. Sakaino, and T. Tsuji: “A cooking support system with force visualization using force sensors and an RGB-D camera”, International AsiaHaptics conference. Springer, Singapore, 2016.

[14] N. point: “Optitrack”, Natural Point, Inc, 2011.