Vol.2022-CG-187 No.3
2022/9/14

条件付き生成モデルを用いたスケッチベースの都市生成手法

神田 純哉[1,a)]　何 毅[1,b)]　謝 浩然[1,c)]　宮田 一乘[1,d)]

概要：コンピューターグラフィックス (CG) や様々なエンターテイメント作品の高度化に伴い，ユーザの設計意図が反映された大規模な街並みを生成することは，映像制作現場における重要な課題の 1 つである．しかし街並みを構成する建物を個別にモデリングすることは労力がかかり現実的ではない．本研究では，スケッチベースのプロシージャルモデリングを使った効率的な街並みの生成手法を提案する．フレームワークには深層学習の条件付き敵対的生成ネットワークを使用する．まず学習データセットとして，パーリンノイズを用いて建物の高さが 1 つずつランダムに計算された街並みを生成する．次に街並みの輪郭を Holistically-nested edge detection (HED) によって抽出する．学習には，街並みのデータから作成したハイトマップと HED によって抽出されたスケッチのペアを用いる．上述のフレームワークを実現するユーザインタフェースを開発し，ユーザが手描きのスケッチから多様で満足度の高い街並みを生成できるようにする．

Sketch-based City Generation Using Procedural Modeling and Generative Model

Abstract: In this study, we propose an efficient city-generation method based on user sketches that combine a deep generative model with the procedural modeling approach. The proposed framework adopts the deep learning-based network of conditional generative adversarial networks. For the data training, we randomly generated three-dimensional cities from perlin noise. The contours of the cities were extracted by the holistically-nested edge detection approach. The proposed method is a deep learning model that uses paired data of cities generated by a procedural model, along with the corresponding hand-drawn style sketches.

1. 序論

大規模な街並みを作成することは，様々なヒューマンコンピューターインタラクションや CG のアプリケーションにおいて必要不可欠である．そのため，実際の制作現場ではクリエーターの負担を軽減することが重要になる．街並みの生成には文法に従って自動的に形成パラメーターが調整されるプロシージャルモデリングが頻繁に用いられる．Townscaper[1] にもそのアルゴリズムが使われており，ユーザはインターフェース上で街並みの生成を楽しむことができる．しかしプロシージャルモデリングではユーザが直接モデリングするわけではないため，同じような特徴を持った生成物が作成されやすく，また希望した通りのコンテンツになるか試行錯誤が必要になる．それに対しスケッチに基づくインターフェースではユーザの設計意図を反映させ，直感的にモデリングすることが可能である．特に深層学習に基づくスケッチインターフェースは，様々なグラフィックデザインに対して有用であるとする研究が報告されている [2], [3], [4].

そこで我々は文法に基づくプロシージャルモデリングとスケッチインターフェースを用いた街並みの生成フレームワークを提案する．提案手法のコアは，プロシージャルモデリングによって生成された街並みと，それに対するスケッチ画像を学習ペアとして深層学習を行ったことである．本研究では，プロシージャルモデリングの一例として Townscaper を用いる．提案するインターフェースを使うことで，ユーザはスケッチから新しい街並みを作ることができる．ユーザは街並みを真上から見た様子をイメージし，スケッチによって建物の場所と高さを指定できる．

[1] 北陸先端科学技術大学院大学
Japan Advanced Institute of Science and Technology (JAIST), Ishikawa, Japan
[a)] jkanda@jaist.ac.jp
[b)] s2010035@jaist.ac.jp
[c)] xie@jaist.ac.jp
[d)] miyata@jaist.ac.jp

本研究では街並みを直感的に生成するスケッチインターフェースを提案し，ユーザがスケッチから街並みをデザインすることを可能にしている．スケッチと深層学習を組み合わせることで，ユーザは容易に大規模な街並みを生成することができ，建物を 1 つ 1 つモデリングする手間から解放され街並みの大局的なデザインに集中することが可能になる．提案するインターフェースによってユーザは詳細なモデリングをする必要がなくなるため，負担を軽減することができる．

2. 関連研究

2.1 スケッチインタフェース

スケッチに基づくモデリングは CG の分野の重要な研究テーマの一つであり，多くの研究者がこの分野で成果を発表している [5]．この研究テーマは CG だけでなくコンピュータービジョンの分野でも同様に重要である．He ら [2] は幾何学的サンプリングを用いてユーザのスケッチから法線マップを生成するための生成モデルを提案している．また，Guerin ら [3] は，スケッチと条件付き敵対的生成ネットワーク (cGAN) を使いインタラクティブな地形生成を可能にした．制作過程において，ユーザはまず川，谷，尾根などの地形を示すスケッチを入力する．事前に学習されたアルゴリズムによって形成パラメーターが計算され，スケッチに対応した地形が自動生成される．さらに浸食による地形の変化も非常に低い計算コストで反映させることも可能にしている．Kelvin ら [6] は cGAN を使った道路網の生成フレームワークを提案し，ゲーム開発に利用している．Han ら [7] は深層学習によって 2 次元のラフなスケッチから 3 次元の顔のモデリングを可能にした．また 3 次元空間にスケッチする 3D スケッチに関する研究成果も多く発表されており，Li ら [8] は 3D スケッチから 3D モデルを探索する最初の試みだったが，単純な形状に限定された．近年では Liu ら [9] は最小 2 本のストロークから 3D の建物を，Yuan ら [10] は 3D の植物を 3D スケッチから生成する手法を提案している．さらに Xing らは [11] はストロークから 3D のヘアースタイルのインタラクティブな探索を可能にして，2D では難しかった複雑なヘアースタイルのモデリング時間を大幅に短縮させた．

2.2 街並み生成

制作意図を反映した大規模な街並みの生成は，多くのヒューマンコンピューターインタラクションや CG のアプリケーションで必要不可欠である．Benes ら [12] は，Urban Brush を開発した．彼らは街は徐々に発展していく様子からインスピレーションを得ており，インタラクティブなブラシの操作によって街並みのレイアウトの一貫性を保ったまま，建物や道路の配置を変更することができる．Emilien ら [13] は画家のパレットのように様々な街並みや地形の傾きを表すブラシを使い分けることで，ユーザの求める街並みの生成手法を提案している．Ritchie ら [14] は Sequential Monte Carlo (SMC) を改良してプロシージャルモデリングの出力を制御する方法を提案している．Niese ら [15] が提案した手法では街並みのレイアウトに合わせて自然に植物を配置させることに成功している．GAN を用いた街並みの生成に関する研究では，Kim ら [16] が 1 枚のストリートビュー画像や写真から 3D 都市を生成する方法を提案している．このように実世界のデータから建物を構築するデータ駆動型のモデリング手法も近年多く見られる．Huang ら [17] は屋根は航空画像から，窓やドアのような側面情報は地上画像から得て建物全体を再構築を可能にし，Zhu ら [18] は斜めの航空写真から大規模な街並みを再現する手法を提案している．Schwärzler ら [19] の提案した手法では点群データと画像データを利用し，2D のサポート線をなぞることで 3D の建物を復元できる．Kelly ら [20] はデータソースとなる画像のノイズを修正するモデルを作成している．しかしこれらの手法は実在する建物の再構築に限定され，ユーザがイメージした街並みを生成することはできない．スケッチベースの街並み生成の研究では，Nishida ら [21] は機械学習を用いることでスケッチが意図する街並みの生成ルールを探索するという手法を提案している．[22] でスケッチベースのモデリングに関する大規模な調査結果が報告されている．

本研究では GAN を用いてスケッチから街並みを自動生成する手法を開発した．ユーザはラフなスケッチを描くだけで魅力的な街並みを容易に生成することができる．

3. 研究背景

Wave Function Collapse Algorithm (波動関数崩壊アルゴリズム）[23] は Townscaper のようなプロシージャルモデリングを利用した街並み生成で用いられる．波動関数とは量子の状態を表す方程式である．確率でしか表すことのできない量子の状態が，観測によって確定されることを波動関数崩壊という．つまり任意の位置の状態を決定するとルールに従って近傍の状態が決まるアルゴリズムのことである．Townscaper では建物を内包するボクセルを隣接する 6 方向の任意の向きから繋げていくと，空中に配置させた建物の屋根は通路に変化し，建物同士が隣接せず接近した場合や建物が空中にせり出した場合には自動的に梯子が架橋される．このように一定性を保ったまま柔軟な補完が繰り返し行われ，街並みを不自然ではない形で発展させていくことが可能になる．このアルゴリズムによってユーザはインタラクティブ且つ容易に既存の街並みを変更し，街並みの範囲も拡張できる．

3.1 プロシージャルモデリング

Townscaper のプロシージャルモデリングでは，まずマ

ウスが置かれている 2 次元座標上のマス目（データ上は corner と表記される）に立方体状のボクセルを設置する．設置したボクセルには水平・垂直のどの方向からもボクセルを追加し，接続させることが可能である．ボクセルの色は 15 色のパレットから選択することができる．このようにマス目，高さ，色のパラメーターは制御パラメーターとして保存される．我々の実装では XML ファイルに保存した (**図 1**)．

3.2 制御パラメーター

本研究で使用したプロシージャルモデリングの制御パラメーターは，各マスの 2 次元座標 (x, y) とそのマスのボクセル数 n，そしてそのボクセルの階層情報 h, t があり，階層情報 h と t は必ず対になったセットの情報として保管されている．なお $h = 0$ は土台を表し (図 1 左)，階層情報にはボクセルの色インデックス t と水面からの高さ h が入力されている．ボクセル数はその階層情報と関連している．図 2 に建物の制御パラメーターの例を示す．左側の建物の制御パラメーターは，$n_l = 1, h_l^1 = 0$(上付きの数字は高さ h を，右辺はボクセルの色インデックス t を表す) であり，右の建物は，$n_r = 2, h_r^1 = 0, h_r^2 = 1$ である．なお，n 個のボクセルはそれぞれ独立した高さ値 $h^i, i \in [1, n]$ を持つ必要があり，ボクセルがない $(n = 0)$ 場合 2 次元座標の情報は保管されずマップ上は水面になる．

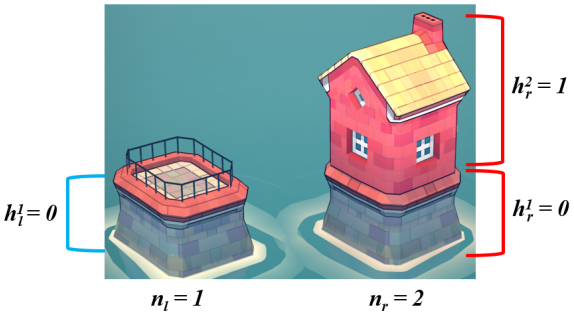

図 1 使用したプロシージャルモデリングで用いられる制御パラメーター．

4. 提案手法

本研究では**図 2** に示すようなフレームワークを提案する．まず，制御パラメータから高さ情報を持つハイトマップを作成する．ハイトマップの視点は固定し，ユーザが都市を真上から見ているものとする．得られたハイトマップに対し Holistically-Nested Edge Detection (HED) 処理を行い，スケッチ風のハイトマップに変換し，それらをデータセットとして深層学習を行った．スケッチに基いたハイトマップの生成には cGAN[24] を使い，スケッチに対応する制御パラメーターを得た．

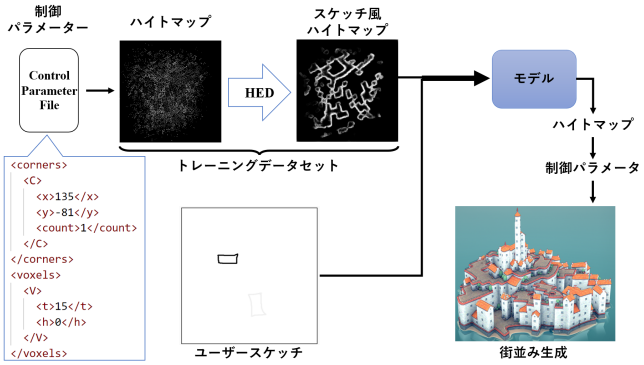

図 2 提案するシステムの概要図．

5. システム実装

5.1 ハイトマップ

まず視点を真上に固定し，見えているすべてのマスを建物のない土台に設定した．次に，それらすべてのマスにパーリンノイズ処理を行い，ランダムな高さを持つ街並みを生成した．パーリンノイズは乱数の程度を制御でき，滑らかに値を変化させることも可能なノイズである．パーリンノイズは引数の値の差が大きいほど，出力される値も 0〜1 の間で差が大きくなる．そこで引数に掛ける値をノイズ値とし，出力される値の差を調整した．各マスの建物の高さはパーリンノイズの出力値となるため，ノイズ値が大きくなるほどマスごとに違った高さの建物が生成される．偏りなく様々な高さの建物のデータを得るためにノイズ値は十分に大きく設定し，200.0 とした．また最終的にスケッチから街並みを生成することを考慮して建物の最大高さを設定する必要がある．Townscaper の仕様上の建物の最大高さは 255 階である．そのまま最大を 255 階にしてしまうと街並みを構成する前方の建物しか見えなくなり，一目で意図した街並みが生成されたかどうか判断できなくなる．反対に最大を 3 階程度にしてしまうと生成物のバリエーションが無くなってしまう．生成される街並みを容易に確認でき，さらにユーザの自由度を確保できる高さとして最大を 14 階とした．次に，制御パラメータの座標情報が，視点を固定した 256 × 256 画素のマップ上のどこに対応しているかを調べた．

その方法を**図 3** に示す．まず視点を固定したマップ上で対象とするマスを 1 つ決め，図の左上のように建物がそのマスのみに 1 つだけ表示されている画像を得る．次にその画像をグレースケール画像に変換し二値化する．建物のみが二値化の対象となるように閾値は 169 とした．二値化処理後の RGB 値が 255 の座標を調べ，制御パラメータの座標情報と対応させた．これを視点を固定したマップ上の全てのマスに対して行い，制御パラメータ内の座標情報をハイトマップ内の座標情報に変換した．最後に制御パラメータの高さ情報とハイトマップの座標情報を統合し，最終的

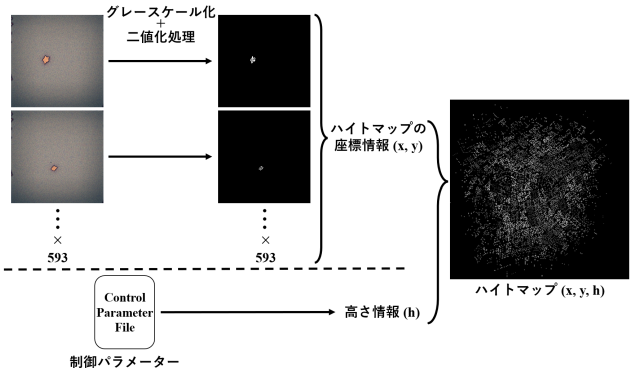

図 3 ハイトマップの作成手順．視点が固定されたマップ上には 593 個のマスがある．そのためグレースケール化+二値化処理を 593 回行った．

なハイトマップを得た．

5.2 ネットワーク

ネットワーク構造は 9 ブロックの ResNet ベースの生成器 G と，pix2pix と呼ばれる GAN を利用した画像生成アルゴリズムの既存設定である PatchGAN 識別器 D を適用している (**図 4**)．まず，p_s，p_{gt}，p_g はそれぞれスケッチ，ハイトマップのグランドトゥルース，生成されたハイトマップの領域の分布に定義する．ネットワークは入力スケッチ $x \in p_s$ とランダムノイズ z からハイトマップ $y \in p_{gt}$ へのマッピングを学習する．その過程は $G : x, z \rightarrow y$ で表す．ネットワークは $loss = loss_g + w_{l1} * loss_{l1} + w_{hm} * loss_{hm}$ と定義される損失関数で学習される．ここで，w_{l1}，$loss_{l1}$，w_{hm}，$loss_{hm}$ はそれぞれ L1 損失の重み，L1 損失，ハイトマップ損失の重み，ハイトマップ損失を意味する．また，$loss_g$（式 1）は cGAN の訓練データにおける損失を表す．L1 損失（式 2）とハイトマップ損失（式 3）は生成結果とグランドトゥルースの高さ分布をより一致させるために使用される．式 3 では，生成されたハイトマップの高さ h の値の合計を制限している．

$$loss_g = E_{x\sim p_s, y\sim p_{gt}}[logD(x,y)] + E_{x\sim p_s}[log(1 - D(x, G(x,z)))]\,. \quad (1)$$

$$loss_{l1} = E_{y\sim p_g, \tilde{y}\sim p_{gt}}\left[\|y - \tilde{y}\|_1\right]\,. \quad (2)$$

$$loss_{hm} = |\sum_{i\in y\sim p_{gt}} h_i - \sum_{j\in \tilde{y}\sim p_g} h_j| \quad (3)$$

5.3 ユーザインターフェース

スケッチするための**図 5** のようなユーザインタフェース（UI）を Vue.js で開発し，Web アプリケーションとして利用できるようにした．またバックエンドシステムは Flask で実装した．UI では描画，部分消去，作業を 1 つ戻す又は進める，保存の機能が実装されている．また，建物の高さを指定できるように線のグレースケールを調整することができる．濃い線は高い建物を，薄い線は低い建物を指定する．なお今回の UI では建物の色は自動的に白色になる．

6. 結果と考察

提案システムは，NVIDIA RTX 3090 グラフィックスカード，Intel(R) Xeon(R) W-2223 CPU，64GB メモリを搭載した Linux システムコンピュータ上に実装された．ネットワークは PyTorch で実装した．ネットワークの学習には，最適化として確率的勾配降下を用いた．w_{l1} と w_{hm} はそれぞれ 100 と 0.001 に設定した．

6.1 街並み生成

このシステムによって生成された街並みを**図 6** に示す．このインタフェースでは，黒い線は高い建物を，明るい線は低い建物を指定することができる．インターフェースに入力されたスケッチから生成されるのが左から 2 番目のハイトマップである．ハイトマップには 256 × 256 画素分の高さ情報が入力されているため，その情報を制御パラメーターの座標情報と高さ情報に変換したのが左から 3 番目のハイトマップである．制御パラメーターの 1 マスに対し複数の高さ情報が含まれるため，最も出現頻度の多い高さを制御パラメーターの高さ情報とした．単純なスケッチの場合，おおむね意図を反映した街並みが生成されている．最後のスケッチの例では，グレースケールの色の使い方を変え，建物の高さも変えている．

6.2 ユーザスタディ

今回開発したシステムを使うことで，ユーザがスケッチから容易に都市を生成できるか評価するために，ユーザ調査を実施した．調査対象者は 23〜28 歳の 5 名 (男性 4 名，女性 1 名) とした．まずユーザには 5 分程度で簡単な UI の説明を行い，2 つの注意点を伝えた．1 つ目は上から見た街並みをイメージし，グレースケールの濃淡を使って建物の高さを調整すること，2 つ目はスケッチでは建物の境界を細かく指定できないことである．ユーザには 45 分以内に 5 枚のスケッチを作成してもらい，アンケートによりユーザ体験を評価した．アンケートには System Usability Scale (SUS) を用い，1 を「強く反対」，5 を「強く賛成」とした．具体的な内容および得点を**表 1** に示す．調査の最後に参加者から 5 分以内でフィードバックを得た．

SUS の結果は最高点が 70 点，最低点が 50 点，平均が 63.5 点であった．Bangor ら [25] は平均点が 70 点以上を評価できる点数と考えており，それを踏まえると今回のインタフェースは良いとは判断できないことを表している．各質問に対する点数を比較すると，質問 2 と 5 の得点が高く，質問 8 の得点が低くなっている．質問 2，5 は使いやすさ

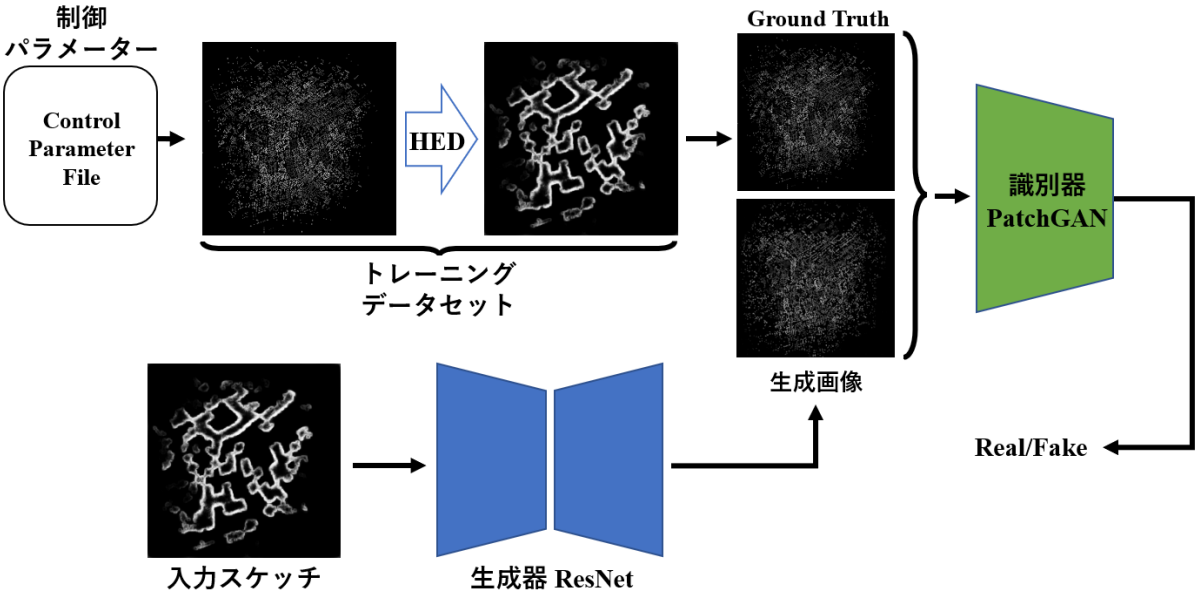

図 4 提案システムのネットワーク構造．制御パラメータから作成されたハイトマップは学習データであり，識別器の入力として用いられる．

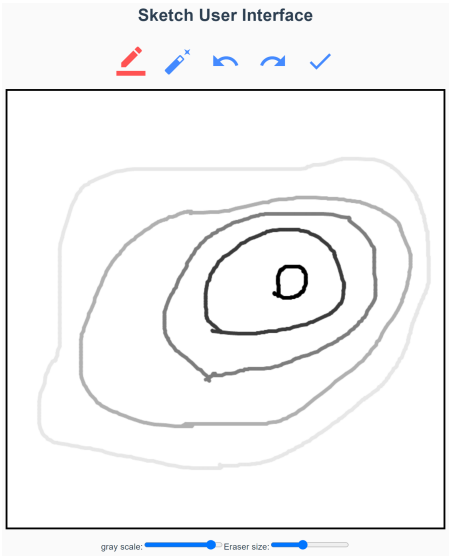

図 5 実装したユーザインターフェース．上部のメニューアイコンは，左から描画するブラシ，消しゴム，作業を 1 つ戻す，作業を再び進める，セーブの機能を表す．下部のバーではグレースケールの濃淡や消しゴムの大きさを調整することができる．

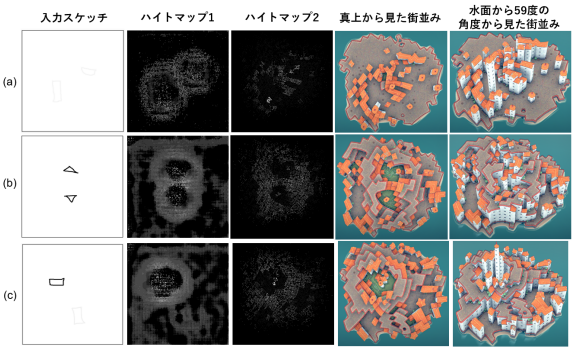

図 6 スケッチから生成される街並みの例．左から入力スケッチ，ハイトマップ 1(スケッチから生成されたハイトマップ)，ハイトマップ 2(制御パラメーターのマスに対応させたハイトマップ)，生成された街並みを真上から見た図，街並みを水面から 59 度の角度から見た図．(a) 薄い線で低い建物を生成した結果．(b) 濃い線で高い建物を生成した結果．(c) 濃淡を組み合わせてスケッチした結果．角度を変えて観察すると，さまざまな高さの建物を生成できていることが分かる．

を問う項目である．このことから，提案するインターフェイスはシンプルで容易に街並みを生成できていることが伺える．一方，質問 8 の結果は生成された街並みがユーザの意図をうまく反映できていないことを示唆している．これは，事前に想定された以上にユーザは詳細なスケッチを好んだため，その意図を全て反映させられなかったことが原因であると考える．そのため，調査終了後に感想を聞いたところ「自分のイメージと違う街になっていた」，「高低差が分かりにくかった 」という声もあった．5 枚スケッチし街並みを生成するのにかかった時間は最長で 45 分，最短で 35 分だった．スケッチのストローク数は 1 番多いスケッチで 36 本，1 番少ないスケッチで 3 本になり，全体の平均は 10.24 本だった．またユーザがスケッチした 25 枚の内 3 枚は線を引くだけでなく塗り潰したスケッチになった．

6.3 考察

ユーザーが描いたような複雑なスケッチの例を**図 7** として示す．等高線のように円の中心ほど高い建物を生成しようとしたが，必ずしもそのような街並みは形成されていない．さらに全くスケッチされていない左下の部分にも建物が生成されてしまっており，スケッチした線の影響する範囲が広く，細かい部分まで指定することは困難なことが分かる．その原因として学習ペアの HED 画像の輪郭が，ハイトマップの高さ情報を全て認識できず，ぼやけてしまっ

表 1 アンケートの質問内容とスコア (平均値と標準偏差).

番号	質問内容	スコア
1	街並みを生成するためにスケッチシステムをしばしば使いたいと思う	2.6 (1.14)
2	スケッチシステムを使用するには説明が必要になるほど複雑であると感じた	3.8 (0.45)
3	スケッチシステムにより多様な都市を作ることができると思う	2.4 (1.14)
4	スケッチシステムを利用するには，専門家・技術者のサポートが必要だと思う	2.8 (1.30)
5	スケッチシステムを使うことで楽に都市を作ることができる	3.8 (0.45)
6	スケッチシステムには一貫性が無いところが多いと感じた	2.2 (1.64)
7	ほとんどの人がスケッチシステムをすぐ使いこなせるようになると思う	2.4 (1.14)
8	スケッチシステムでは思い通りに都市を作ることができないと感じた	1.2 (0.84)
9	スケッチシステムに満足している	1.8 (1.30)
10	スケッチシステムを使いこなすには事前に沢山のことを学ぶ必要があると思う	2.4 (0.55)

ていることにあると考えられる．

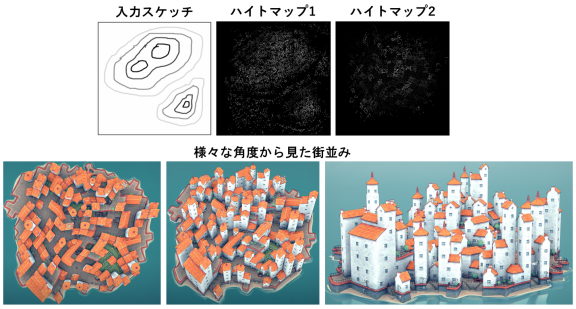

図 7 複雑なスケッチの例．意図した高低差が実現されず，位置関係も対応していない．

これらの課題を踏まえモデルの改良を行った．パーリンノイズを用いて作成した 1 枚の学習データの中には 1〜14 階の建物が含まれている．現在の手法では一度の HED 処理で 1〜14 階の全ての輪郭を取ろうとしているが，このプロセスの難易度が高いと考えられる．そこで 1 枚の学習データを階ごとに分割し，1 階のみの画像，2 階のみの画像，3 階のみの画像といったように 14 枚に分ける．分割した状態で輪郭を取りその後で統合させるアプローチを追加することで輪郭抽出の精度向上を狙った．実装にあたり以下の点を変更した．第 1 に制御パラメーターの高さ情報をハイトマップに変換させる際に，土台のみの場合の RGB 値を 127，最も高い 14 階を 255 とし，間の階は等分に割り振った．こうする事でより輪郭を取りやすくした．第 2 に輪郭の抽出方法を HED からモルフォロジー変換に変更した．HED によって輪郭を抽出すると，統合させる際に輪郭部分が重なってしまい学習データとして使うことが出来なくなったためである．第 3 に分割したデータを統合させる際は 14 階分全てを使うのではなく，ランダムに 3 つの階を選択し学習データとした．学習データ量を調整したところ，最もスケッチの意図が反映されたのは 3 階分だった．第 4 にハイトマップから制御パラメーターに変換する際に採用する高さを最頻値から平均値に変更した．現在のモデルでは図 6 に示すようにスケッチの影響が広範囲に広がってしまっている．その影響を緩和させるため制御パラメーターの高さ情報として最頻値を使っていたが，新しいモデルではこのような現象はなくなったため平均値の方が適切だと判断した．

以上のような変更を行って新たにモデルを作成した．新たなモデルを使いスケッチして街並みを生成した結果を**図 8** として示す．(a) と (b) を見ると以前のモデルと異なりスケッチした部分のみに街並みが生成され，特に (b) のように複雑なスケッチをした場合でも意図が反映されている．しかし (c) のようなスケッチのパターンだと上手く生成されていない．この原因は学習データにあると考えられる．学習に使用されている建物の高さはランダムに決められており，1 マス 1 マスが独立している．そのため基本的に広範囲に渡って同じ高さの建物が続くということはなく，また山のように一部分が高い状態からなだらかに高さが変化していくといったデータは学習していない．そのため広範囲のスケッチや何重にも線が重複する等高線のようなスケッチに対して対応できないと推測される．

最後に改良前後のモデルの性能差を**図 9** として示す．濃淡を使い分け，同一のスケッチから街並みを生成した．旧モデルを使った (a) ではスケッチした通りのハイトマップや街並みが生成されていない．それに対し新モデルを使った (b) ではスケッチで囲った枠の中にのみ濃淡に応じた高さのハイトマップが作成され，生成された街並みにも意図しない建物は建っていないことが分かる．

7. 結論

本研究では，フリーハンドスケッチから魅力的な街並みを生成するためのフレームワークを提案した．このネットワークを使うことで，ユーザは建物の位置や高さを自由にデザインすることが可能である．

今後等高線のようなスケッチにも対応できるように学習データを用意する必要がある．また三角や四角をスケッチした際に，スケッチ通りのハイトマップを作成できても生成される街並みは若干形が変わってしまう．この原因は制御パラメーターのマス目の解像度にあると考えられる．上

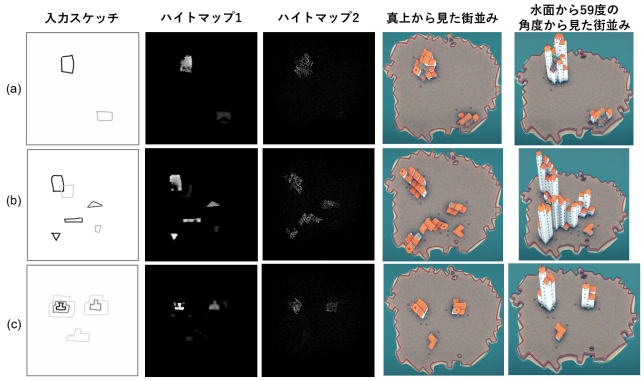

図 8 改良したモデルを使ってスケッチから街並みを生成した例．(a) 濃淡を組み合わせてシンプルにスケッチした結果．(b) 複雑なスケッチの結果．(c) 凸字を等高線のように三重枠，二重枠，単枠で囲った結果．

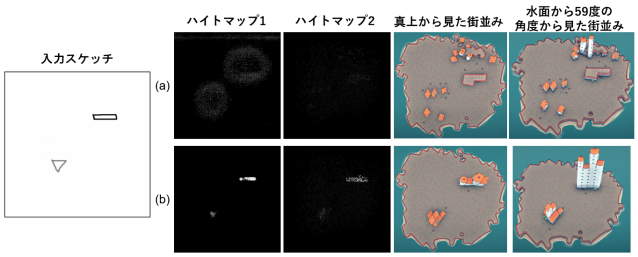

図 9 同一のスケッチから改良前と改良後のモデルを使って街並み生成の過程を比較した．(a) 改良前のモデルを使って街並みを生成した結果．(b) 改良後のモデルを使って街並みを生成した結果．

述したように視点が固定されたマップ上には 593 個のマスがある．視点を離してマスの数を増やせば更に解像度を上げることが出来るため，スケッチ通りの形の街並みを生成できると期待される．また UI にマス目を見えるようにしておくことで，ユーザーは生成される街並みをイメージしながらスケッチ出来るため今後実装したい．また現在のシステムでは白色の建物を用いた街並みしか生成できず，スケッチも真上からの視点に限定される。そのため色を指定するブラシを追加し，様々な角度からスケッチできるようにして自由度を高める UI の開発を検討している．さらなる展開として，他のプロシージャルモデリングへの Wave Function Collapse Algorithm の応用や，衛星画像を学習データセットとすることでスケッチから現実世界の街並みを再構築することも考えている．

謝辞

本研究の一部は，北陸先端科学技術大学院大学研究拠点形成支援事業および科研費 20K19845 の支援により実施された．

参考文献

[1] Oskar Stålberg. Townscaper. 2020.
[2] Yi He, Haoran Xie, Chao Zhang, Xi Yang, and Kazunori Miyata. Sketch-based normal map generation with geometric sampling. In *International Workshop on Advanced Imaging Technology (IWAIT) 2021*, Vol. 11766, p. 117661B. International Society for Optics and Photonics, 2021.
[3] Éric Guérin, Julie Digne, Eric Galin, Adrien Peytavie, Christian Wolf, Bedrich Benes, and Benoît Martinez. Interactive example-based terrain authoring with conditional generative adversarial networks. *Acm Transactions on Graphics (TOG)*, Vol. 36, No. 6, pp. 1–13, 2017.
[4] Zhongyuan Hu, Haoran Xie, Tsukasa Fukusato, Takahiro Sato, and Takeo Igarashi. Sketch2vf: Sketch-based flow design with conditional generative adversarial network. *Computer Animation and Virtual Worlds*, Vol. 30, No. 3-4, p. e1889, 2019. e1889 cav.1889.
[5] Luke Olsen, Faramarz F Samavati, Mario Costa Sousa, and Joaquim A Jorge. Sketch-based modeling: A survey. *Computers Graphics*, Vol. 33, No. 1, pp. 85–103, 2009.
[6] Lin Ziwen Kelvin and Bhojan Anand. Procedural generation of roads with conditional generative adversarial networks. In *2020 IEEE Sixth International Conference on Multimedia Big Data (BigMM)*, pp. 277–281. IEEE, 2020.
[7] Xiaoguang Han, Chang Gao, and Yizhou Yu. Deepsketch2face: a deep learning based sketching system for 3d face and caricature modeling. *ACM Transactions on graphics (TOG)*, Vol. 36, No. 4, pp. 1–12, 2017.
[8] Bo Li, Yijuan Lu, Azeem Ghumman, Bradley Strylowski, Mario Gutierrez, Safiyah Sadiq, Scott Forster, Natacha Feola, and Travis Bugerin. 3d sketch-based 3d model retrieval. In *Proceedings of the 5th ACM on International Conference on Multimedia Retrieval*, pp. 555–558, 2015.
[9] Zhihao Liu, Fanxing Zhang, and Zhanglin Cheng. Buildingsketch: Freehand mid-air sketching for building modeling. In *2021 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pp. 329–338. IEEE, 2021.
[10] Qi Yuan and Yongjian Huai. Immersive sketch-based tree modeling in virtual reality. *Computers & Graphics*, Vol. 94, pp. 132–143, 2021.
[11] Jun Xing, Koki Nagano, Weikai Chen, Haotian Xu, Li-yi Wei, Yajie Zhao, Jingwan Lu, Byungmoon Kim, and Hao Li. Hairbrush for immersive data-driven hair modeling. In *Proceedings of the 32Nd Annual ACM Symposium on User Interface Software and Technology*, pp. 263–279, 2019.
[12] Bedrich Benes, Xiaochen Zhou, Pascal Chang, and Marie-Paule R Cani. Urban brush: Intuitive and controllable urban layout editing. In *The 34th Annual ACM Symposium on User Interface Software and Technology*, pp. 796–814, 2021.
[13] Arnaud Emilien, Ulysse Vimont, Marie-Paule Cani, Pierre Poulin, and Bedrich Benes. Worldbrush: Interactive example-based synthesis of procedural virtual worlds. *ACM Transactions on Graphics (TOG)*, Vol. 34, No. 4, pp. 1–11, 2015.
[14] Daniel Ritchie, Ben Mildenhall, Noah D Goodman, and Pat Hanrahan. Controlling procedural modeling programs with stochastically-ordered sequential monte carlo. *ACM Transactions on Graphics (TOG)*, Vol. 34, No. 4, pp. 1–11, 2015.
[15] Till Niese, Sören Pirk, Matthias Albrecht, Bedrich Benes, and Oliver Deussen. Procedural urban forestry. *ACM Transactions on Graphics (TOG)*, Vol. 41, No. 2, pp. 1–18, 2022.

[16] Suzi Kim, Dodam Kim, and Sunghee Choi. Citycraft: 3d virtual city creation from a single image. *The Visual Computer*, Vol. 36, No. 5, pp. 911–924, 2020.

[17] H Huang, M Michelini, M Schmitz, L Roth, and H Mayer. Lod3 building reconstruction from multi-source images. *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences*, Vol. 43, , 2020.

[18] Lingjie Zhu, Shuhan Shen, Xiang Gao, and Zhanyi Hu. Large scale urban scene modeling from mvs meshes. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[19] Michael Schwärzler, Lisa-Maria Kellner, Stefan Maierhofer, and Michael Wimmer. Sketch-based guided modeling of 3d buildings from oriented photos. In *Proceedings of the 21st ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, pp. 1–8, 2017.

[20] Tom Kelly, John Femiani, Peter Wonka, and Niloy J Mitra. Bigsur: large-scale structured urban reconstruction. *ACM Transactions on Graphics*, Vol. 36, No. 6, 2017.

[21] Gen Nishida, Ignacio Garcia-Dorado, Daniel G Aliaga, Bedrich Benes, and Adrien Bousseau. Interactive sketching of urban procedural models. *ACM Transactions on Graphics (TOG)*, Vol. 35, No. 4, pp. 1–11, 2016.

[22] Sukanya Bhattacharjee and Parag Chaudhuri. A survey on sketch based content creation: from the desktop to virtual and augmented reality. In *Computer Graphics Forum*, Vol. 39, pp. 757–780. Wiley Online Library, 2020.

[23] Maxim Gumin. Wave Function Collapse Algorithm, 9 2016.

[24] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, 2017.

[25] Aaron Bangor, Philip Kortum, and James Miller. Determining what individual sus scores mean: Adding an adjective rating scale. *Journal of usability studies*, Vol. 4, No. 3, pp. 114–123, 2009.