2Y-08

手描きアニメーションの背景画からの投影変換関数の決定法

山根初美　岡田大樹　齋藤豪
東京工業大学 情報理工学院

1　はじめに

手描きアニメーションのレイアウト図で使われている透視図法から投影変換関数を求めることができれば，背景の中のキャラクターの配置やカメラの姿勢などをインタラクティブに加工・調整することが可能になり，アニメーション制作におけるレイアウト工程の支援につながることが期待できる．

しかし，手描きアニメーションに用いられる図法では，空間を広く見せることなどを目的に消失点が 1 つに定まらないような技法が用いられることがあり，通常の透視投影を前提とする既存手法では投影変換関数を再構成することが不可能な場合がある．

このような図法は，擬似多視点投影として解釈できる．Yoshimura ら [1] が提案した統合投影法では，投影を制御するために 4 つの値，カメラの画角 FoV, x 方向のせん断量 α, y 方向のせん断量 β, 投影参照面と呼ぶスクリーンと平行なカメラからある距離にある面からの距離に応じて y 方向に空間を伸び縮みさせるための値 s を用いることで，ビューボリュームの上下左右の 4 面をそれぞれ独立して動かせるようにし，擬似多視点投影を実現する．

本稿では，これら α, β, s および FoV の代わりとして焦点距離 f を推定する手法と，ユーザが背景画像から対話的にこれらの値を決定できるユーザインタフェースを提案する．

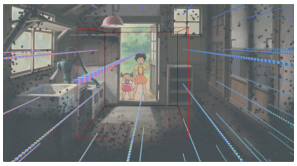
2　提案手法

ユーザは，背景画に描かれた 3 次元空間に対して，その中に置いた立方体の投影後の図をスクリーン上に描画する（図 1）．描かれた立方体から 4 つの制御変数を推定し，投影変換関数を決定する．統合投影法のスクリーンと投影参照面の原点からの距離 N, λ は固定値とし，これらは一致するものとする．スクリーンの横幅 W，縦幅 H は入力された背景画から決定する．

2.1　ユーザによる入力立方体とその補正

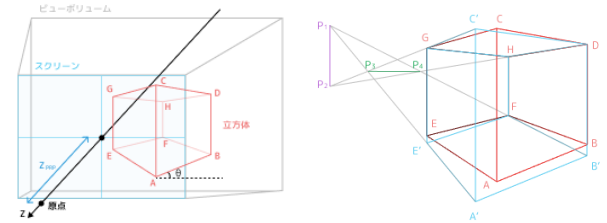
図 2 (a) に示すように，ユーザが描画する立方体は，地面（xz 平面）に対して水平になるように置かれ，また，鉛直方向の辺のうち少なくとも 1 つはスクリーンと同じ z 座標に位置し，側面の 1 辺とスクリーンのなす角が θ であるものとする．8 つの頂点を画面上で移動させることで，ユーザは立方体図形を入力する．

(a) 背景画像 [2]　(b) ユーザ入力



(c) 推定投影関数により投影した小立方体との合成

図 1: 投影変換関数の推定法の利用例



(a) 3 次元空間内に置いた立方体　(b) 投影後の立方体の補正

図 2: 補助入力

擬似多視点投影は，投影前後で鉛直線が保持されるという性質と，空間の平行線の投影後の交点 (消失点) が水平・鉛直に並ぶという性質がある．そのため，図 2 (b) の立方体が正しく描画されている場合は，以下の 2 つの制約が満たされている．

1. AC, BD, EG, FH が鉛直線である
2. AE と BF, CG と DH，DH と BF，CG と AE の延長線が交わる点をそれぞれ P_1, P_2, P_3, P_4 とおくと，P_1P_2 は鉛直線，P_3P_4 は水平線となる

投影変換関数を適切に推定するには，入力された図形が，これら制約を満たしていることが望ましい．そこで，2 つのユーザ入力像の補正法を導入する．

補正法 1　例としてユーザが頂点 A を点 A' の位置に動かした場合，制約 1，すなわち $C'_x = A'_x$ を満たすように，頂点 C を点 C' の位置まで x 方向に平行移動させる．

Estimation of Projection Function from Single Background Image in Animation
Hatsumi YAMANE
Daiki OKADA
Suguru SAITO
School of Computing, Tokyo Institute of Technology

補正法 2　補正法 1 に加えて，制約 2，すなわち $P_{1x} = P_{2x} \wedge P_{3y} = P_{4y}$ を満たすように，ユーザが動かした頂点の隣接点の他の 2 点の y 座標を調整する．P_1, P_2, P_3, P_4 は，実数 $t_1, \cdots, t_8$ および点 A から H とを用いて，以下のように表せる．

$$P_1 = (1-t_1)\,E' + t_1 A' = (1-t_2)\,F + t_2 B' \quad (1)$$

$$P_2 = (1-t_3)\,G + t_3 C' = (1-t_4)\,H + t_4 D \quad (2)$$

$$P_3 = (1-t_5)\,H + t_5 D = (1-t_6)\,F + t_6 B' \quad (3)$$

$$P_4 = (1-t_7)\,G + t_7 C' = (1-t_8)\,E' + t_8 A' \quad (4)$$

式 (1) から (4) を $t_1, \cdots, t_8$ について解くことで，P_1, P_2, P_3, P_4 の座標を A から H の座標で表すことができる．これを制約 $P_{1x} = P_{2x}$，$P_{3y} = P_{4y}$ に代入し解くと (B'_y, E'_y) の解の組が 2 つ得られ，そのうちの一方を採用する．

ユーザが点 A 以外の頂点を動かした場合も，同様の補正を行う．

2.2　投影変換関数の推定

関数を制御する 4 つの値の推定は，次のように行う．

3 次元空間での立方体の一辺の長さを l とおくと，$l = C_y - A_y$ と表せる．カメラの焦点距離 f，x 方向のせん断量 α，立方体の回転角度（辺 AB が x 軸となす角） θ を，以下の (5) から (7) 式を満たすような解を探索することで求める．

$$(B_x + f\alpha)(f + l\sin\theta) = f(l\cos\theta + A_x + f\alpha) \quad (5)$$

$$(E_x + f\alpha)(f + l\cos\theta) = f(-l\sin\theta + A_x + f\alpha) \quad (6)$$

$$\begin{aligned}&(F_x + f\alpha)(f + l(\cos\theta + \sin\theta)) \\ &= f(l(\cos\theta - \sin\theta) + A_x + f\alpha)\end{aligned} \quad (7)$$

次に，投影参照面からの距離に応じて空間の y 軸方向を伸び縮みさせるための値 s を（8）式，y 方向のせん断量 β を（9）式を解くことで求める．このとき，z_F は入力された立方体に対応する 3 次元空間の頂点 F の z 座標で，$z_F = -f - l\cos\theta - l\sin\theta$ と表せる．

$$H_y - F_y = l(1 + (z_F + f)s)\left(\frac{f}{-z_F}\right) \quad (8)$$

$$f(A_y(1 + (z_F + f)s) + f\beta) = -z_F(F_y + f\beta) \quad (9)$$

3　実験結果と評価

3 名のユーザが，提案手法のインタフェースを用いて，図 3 の背景画に対してそれぞれ 10 回ずつ投影変換関数を推定する実験を行った．推定した α, β, s と f から計算された FoV と元の背景画の投影に用いられている真の値との平均誤差と標準偏差を表 1 に示す．

図 3: 実験に用いた背景画像

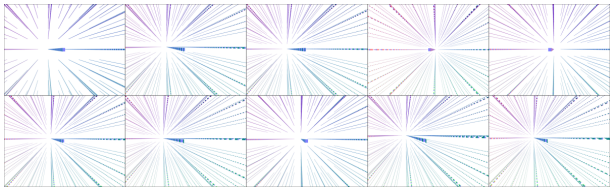


図 4: 推定した投影関数により投影した小立方体

表 1: 推定した投影変換関数の平均誤差と標準偏差

	平均誤差				標準偏差			
	FoV(deg)	α	β	s	FoV(deg)	α	β	s
user 1	13.1	-0.275	0.005	-0.167	20.7	0.282	0.050	0.967
user 2	22.9	0.643	-0.049	0.145	16.5	0.357	0.054	0.126
user 3	8.665	0.007	-0.008	0.005	20.7	0.609	0.029	0.999
all users	14.9	0.125	-0.017	-0.005	20.3	0.464	0.051	0.816

また，空間内に一定間隔で並べた小立方体に対して，ユーザ 1 の入力によって推定したそれぞれの投影変換関数を用いて投影した画像を図 4 に示す．

推定された投影変換関数を用いた投影像はおよそ一致するが，ばらつきも生じている．表 1 より，これらの推定された投影変換関数は，画面縦方向のせん断量 β に比べ，画面横方向のせん断量 α の標準偏差が大きい投影を生じさせることがわかる．よって，図 4 に見られるように，画面縦方向に並んだ奥に向かう平行線の消失先はほぼ一定となるが，画面横方向に並んだ奥に向かう平行線の消失先は振れた投影となる．

4　まとめ

消失点が 1 つに定まらないような図法を含む単体の背景画から，統合投影法の投影変換関数を推定する手法と，対話的に制御変数を決定できるユーザインタフェースを提案した．今後の課題として，推定結果のばらつきを小さくできるよう，ユーザインタフェースと解の探索方法の改良がある．また，本稿で提案した手法ではユーザ入力像のみを用いて投影変換関数の推定を行っているが，背景画像から得られる情報も反映されるよう推定方法を改善することにより，意図した投影変換関数を得やすくなると考えられる．

参考文献

[1] Fujiko Yoshimura and Suguru Saito. Generalized projection for yamato-e and ukiyo-e with projection reference plane. CGI '17, pp. 19:1–19:6, 2017.

[2] となりのトトロ - スタジオジブリ | studio ghibli. https://www.ghibli.jp/works/totoro/, Accessed: 2020-12-12.