

3D Vehicle Reconstruction via Monocular Camera with Deep Learning Models and Direct Linear Transformation

2024-2 Robot Vision (M3228.003000)

Thomas Putzer
Computer Science and Engineering
Seoul National University
Seoul, South Korea
putzer_thomas@snu.ac.kr

Weihao Chao
Mechanical Engineering
Seoul National University
Seoul, South Korea
email address or ORCID

Anggraini Ditha
Civil and Environmental Engineering
Seoul National University
Seoul, South Korea
email address or ORCID

Abstract—Hello here goes the abstract

I. INTRODUCTION

[1] (Leave this citation for now, otherwise it breaks (if we have 0 citations, and the command at the end))

Bla Bla Bla Bla

II. 3D POSE ESTIMATION

($A = U\Sigma V^T$) [2] In the SVD $Ab = U\Sigma V^T b = 0$ implies $V^T b = 0$ since U is invertible and has no nontrivial solutions for $Ub = 0$. Thus $V^T b$ must be a vector where only the rows corresponding to zero singular values contribute to the null space. The columns of V that correspond to zero singular values in Σ combined form a basis for the null space of A .

$$\begin{bmatrix} R & t \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

\leq

```
#Pseudo code of the matching process
#cars keeps track of all the cars in the scene
#cars_n contains all the cars in the new frame
for old_car in cars:
    c, d = closest_car(cars_n, old_car)
    if d > max_distance:
        #the car left the frame
        cars.remove(old_car)
    else:
        old.match(c)
        #every car can only be matched once
        cars_n.remove(new_car)

#all remaining not matched new cars
for new_car in cars_n:
    #a new car has entered the frame
    cars.append(new_car)
```

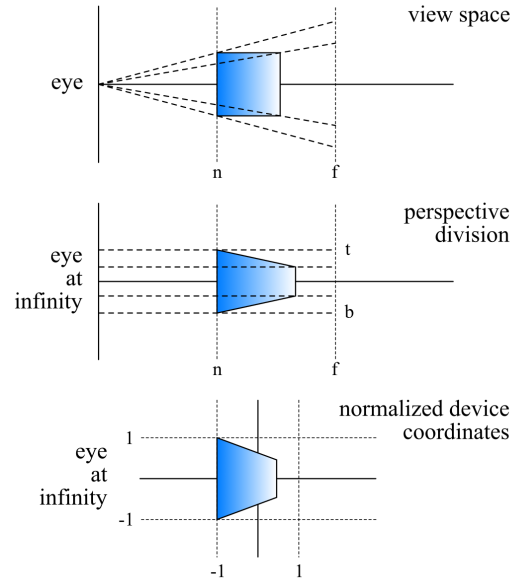


Fig. 1. Perspective projection from view space to NDC

double column text here. . double column text here. .
.double column text here. . .lksaj fundingasdf
. double column text here. . lsakdjf laks icha heise
. double column text here. . laskdjf lasdff cich ic ach .

lkasdjf lakjsd flkajsdl fkjasdl fjasldj flaskdjf laskdjf laksjdfl
kajsdflkajsdflfjasldkfjalsdkjflaskdjf laskdjf lasjkdf aksjdflkasjd
flaskdjf lkasdjf lkasjdfl laksjdfl aksjdflf asldkfj alsdjkf lasdjf
laskdjf laskdjf laskdjfl aksdjfl askdjf askjf l

$$FOV = 2 \cdot \arctan\left(\frac{s_w}{2f}\right), r = n \cdot \tan\left(\frac{FOV}{2}\right) \quad (1)$$

$$M_P = M_{P''} M_{P'} = \begin{bmatrix} \frac{2}{r-l} & 0 & 0 & \frac{-(r+l)}{r-l} \\ 0 & \frac{2}{t-b} & 0 & \frac{-(t+b)}{t-b} \\ 0 & 0 & \frac{2}{f-n} & \frac{-(f+n)}{f-n} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} n & 0 & 0 & 0 \\ 0 & n & 0 & 0 \\ 0 & 0 & n+f & -nf \\ 0 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} c_1 & 0 & 0 & 0 \\ 0 & c_2 & 0 & 0 \\ 0 & 0 & c_3 & c_4 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad (3)$$

$$\begin{bmatrix} -c_1 x_1 & -c_1 y_1 & -c_1 z_1 & -c_1 & 0 & 0 & 0 & 0 & x_1 x'_1 & y_1 x'_1 & z_1 x'_1 & x'_1 \\ 0 & 0 & 0 & 0 & -c_2 x_1 & -c_2 y_1 & -c_2 z_1 & -c_2 & x_1 y'_1 & y_1 y'_1 & z_1 y'_1 & y'_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ -c_1 x_n & -c_1 y_n & -c_1 z_n & -c_1 & 0 & 0 & 0 & 0 & x_n x'_n & y_n x'_n & z_n x'_n & x'_n \\ 0 & 0 & 0 & 0 & -c_2 x_n & -c_2 y_n & -c_2 z_n & -c_2 & x_n y'_n & y_n y'_n & z_n y'_n & y'_n \end{bmatrix} \begin{bmatrix} a \\ b \\ \vdots \\ k \\ l \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix} \quad (4)$$

To get the screen-space position of a vertex, we multiply it by its Modelview Matrix and API Projection Matrix before doing the projective division. We have all the parameters except the Model Matrix. $p' = M_{API} M_{Modelview} p$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} \stackrel{\text{div w}}{\equiv} \begin{bmatrix} c_1 & 0 & 0 & 0 \\ 0 & c_2 & 0 & 0 \\ 0 & 0 & c_3 & c_4 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

If we multiply that out, we get for x' and y' :

$$x' = \frac{c_1 a x + c_1 b y + c_1 c z + c_1 d}{i x + j y + k z + l}$$

$$y' = \frac{c_2 e x + c_2 f y + c_2 g z + c_2 h}{i x + j y + k z + l}$$

We can multiply by the denominator and solve for 0. Since this holds for every pair of points $(p_1, p'_1), (p_2, p'_2), \dots, (p_n, p'_n)$ that we track we can create the following system of equation:

4

REFERENCES

- [1] J.-B. Michel, Y. K. Shen, A. P. Aiden, A. Veres, M. K. Gray, T. G. B. Team, J. P. Pickett, D. Hoiberg, D. Clancy, P. Norvig, J. Orwant, S. Pinker, M. A. Nowak, and E. L. Aiden, "Quantitative analysis of culture using millions of digitized books," *Science*, vol. 331, no. 6014, pp. 176–182, 2011. [Online]. Available: <https://www.science.org/doi/abs/10.1126/science.1199644>
- [2] "Singular value decomposition," https://en.wikipedia.org/wiki/Singular_value_decomposition, accessed: 2024-12-06.