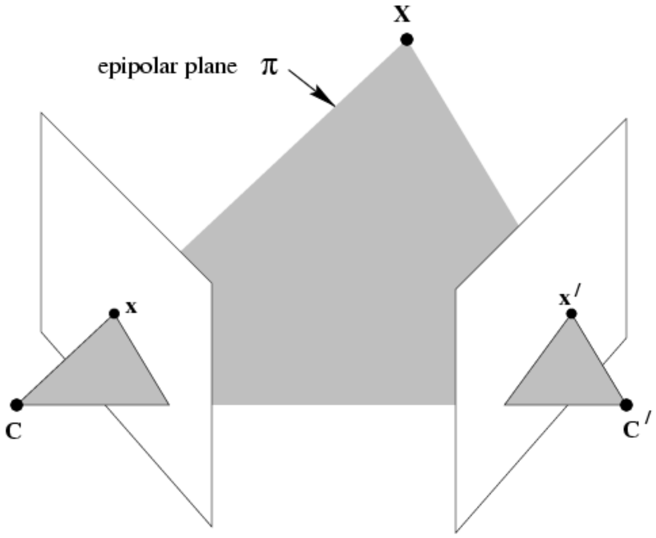
Slide 4 複習 epipolar geometry



所謂 epipolar geometry：

用兩個 camera 拍同一個 scene，所對應的 feature point 對應到同一個 3D 點 X 的兩個 2D 的 feature point：x, x’，他們兩個之間是有一些 constraint 存在，必須滿足以下的關係 $x'^T Fx = 0$ ，F 是 fundamental matrix；x’, x 分別是兩個 camera 在 image plane 所得到的 feature point 的 2D Projection。

它的用途有二：

(1) 可以用這個 fundamental matrix 來驗證我們的 feature tracing 有沒有錯。 $x'^T Fx = 0$

(2) 把 fundamental matrix 當作 projective 的 factorization。 $x'^T Fx \neq 0$

再說清楚一點，因爲我們在拍攝場景的時候，整個 image plane、optical center 跟 3D 中的 scene 的 3D projection 之間有一些 geometry 的關係，而 fundamental matrix 的 equation(x’TFx = 0)的目的就是在於 capture 這個 geometry 之間的關係；用代數的形式來描述幾何的 constraint；值得注意的是，兩個 feature point 之間必須滿足這個 equation 的關係。

Slide 5 Structure from Motion

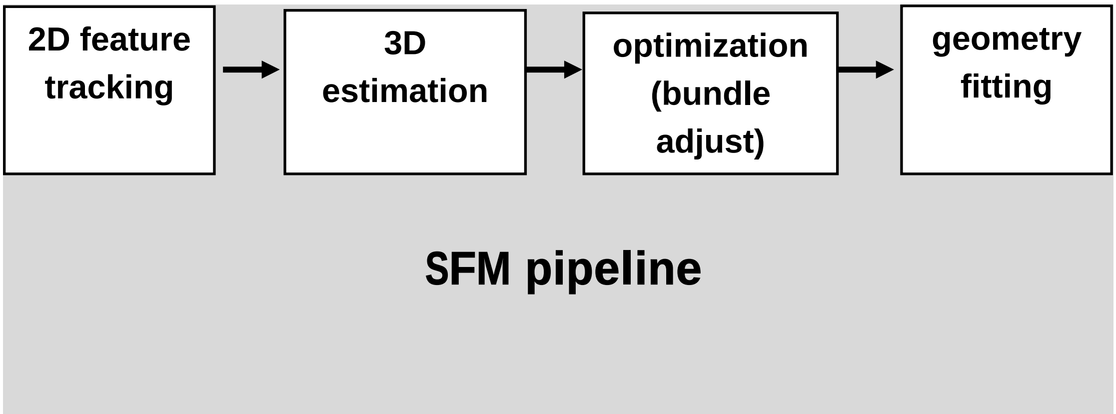
Structure from motion 的問題：

給定 a sequence of images, 理論上可以從這些 sequence of images 找到一些 feature points，然後找出每一個 image 所對應的 camera 的參數以及每一個 feature point 所對 3D 的 projection 應該是什麼。

Input：是一大堆 image sequences。

Output：m 個 camera 分別的 projection matrix，以及 n 個 3D 中的點的 3D 座標值。

Slide 6 Structure from Motion 的流程



Structure from motion 可以分成四個步驟：

(1) 從 image 裡面找出它的 feature point。

(2) 做 3D 的 estimation，利用(1)找出來的 feature point 可以大約的來估計每個 camera 的 view 的 parameter 以及其 3D 中的座標。為什麼需要這個步驟？原則上，我們可以直接利用 2D 的 feature point 去作 bundle adjustment，但是 bundle adjustment 是 nonlinear 的 optimization 問題，這樣的問題我們需要一個很好的 initial point，所以這堂課強調的重點是 factorization(也是 HW3 的重點)，基本上有了 factorization 就可以得到比較的 visualization 給 bundle adjustment，這一步會得到 n 個點的 3D 座標值、m 個 camera 的 parameter 值。Matching move 主要只需要 m 個 camera 的 parameter 值；當然，如果我們還要 structure 的話，我們就可以利用這 n 個 3D 的座標點想辦法去 reconstruct geometry。

(3) 可以直接用 library。

(4) Optional。

Slide 8 SFM 的 Notation

(SFM 的目的就是去 recover n 個 3D 的點和 m 個 view)

q 用來表示 2D 的 feature point。

p 就是 3D 中的座標值。

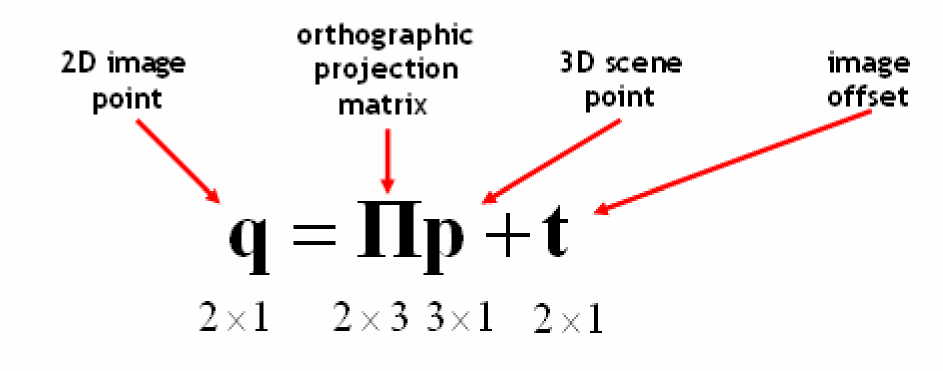
q_{ij}：第 i 個點在第 j 個 image 上面的投影。

λ_{ij}：q_{ij} 這個的 project depth(在 orthogonal 的 case 時我們不會去算到 λ_{ij} ，但是在 projective 的 case 我們就必須去算 λ_{ij})。

$q_{ij} = \pi(\Pi_j p_i)$：給 p_i 這個 3D 的點，經過第 j 個 image 的 projection matrix 的轉換之後，投影到第 j 個 image plane 上，然後經過 projection function，就可以得到 2D 的 feature point，也就是第 i 個點在第 j 個平面上的投影。

z 就是 projective depth。

Slide 9 在 orthographic 投影情況下的 SFM



在 general 的情況下，Π是 3x4 的矩陣，p 是 4x1 的矩陣(x y z 1), q 是 3x1 的矩陣(u v 1)，假設在沒有 translation 的情況下，我們可以把式子改成Π:3x3, p:3x1, q:3x1，把 translation 加到式子的後面就好，又我們在這邊是假設 orthographic projection，所以可以直接拿掉 z 的資訊(把第三個 row，描述 z，拿掉)，因此可以得到 orthographic projection 之下的 equation：Π:2x3, p:3x1, q:2x1。

爲了把 translation 拿走，我們把 n 個 3D 中的點移到他們的 mean vector，也就是說 mean vector 就是這 n 個 3D 點的原點。

當這 n 個 3D 的點，若他們的中心點在原點，把他們投影到任何的 plane 上，這個投影點的中心點，也會是那個 image plane 的原點，因此可以把 t 拿掉，變成 q’ = Πp

Slide 10 Factorization (Tomasi & Kanade)

假設 centroid 是 3D 的原點，2D 的 centroid 是 2D 的原點的話，原則上我們就可以把 translation 拿掉。所以基本上，input 原本是$(u_1, v_1)\ldots(u_n, v_n)$，我們算出的 centroid: $\bar{u}=\frac{1}{n}\sum u_i$ ，$\bar{v}=\frac{1}{n}\sum v_i$ 。將每一個點平移$u_i^{'}=u_i-\bar{u}$ ，$v_i^{'}=v_i-\bar{v}$ 。所以這邊的 $\mathbf{q_1}$ 其實是$(u_1^{'}, v_1^{'})$。 $\underbrace{[\mathbf{q}_1 \ \ \mathbf{q}_2 \ \ \cdots \ \ \mathbf{q}_n]}_{2\times n} = \underbrace{\prod}_{2\times 3} \underbrace{[\mathbf{p}_1 \ \ \mathbf{p}_2 \ \ \cdots \ \ \mathbf{p}_n]}_{3\times n}$

所以我們把所有的 3D 中的點投影到同一個 image 之後，會得到 2*n 這個 **Matrix**，裡面的點是 shift 過後的 2D 的位置所堆疊起來的。而右邊是 shift 過後的 3D 的點所堆疊起來的。再將每一個 view 用這種算法，我們就可以用這種方式來表示:

projection of *n* features in *m* images

$$\underset{2m\times n}{\begin{bmatrix} \mathbf{q}_{11} & \mathbf{q}_{12} & \cdots & \mathbf{q}_{1n} \\ \mathbf{q}_{21} & \mathbf{q}_{22} & \cdots & \mathbf{q}_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{q}_{m1} & \mathbf{q}_{m2} & \cdots & \mathbf{q}_{mn} \end{bmatrix}} = \underset{2m\times 3}{\begin{bmatrix} \mathbf{\Pi}_1 \\ \mathbf{\Pi}_2 \\ \vdots \\ \mathbf{\Pi}_m \end{bmatrix}} \underset{3\times n}{\begin{bmatrix} \mathbf{p}_1 & \mathbf{p}_2 & \cdots & \mathbf{p}_n \end{bmatrix}}$$

W measurement **M** motion **S** shape

左邊的是 2m*n 的 matrix，叫做 measurement matrix，m 代表的是 m 個 view 或 m 個 image，而 n 代表的是 n 個 3D 座標的點。而 $\prod$ 是 projection matrix。而 measurement matrix 是已知的，不過並不是 full rank 的，因爲它必須要符合一些關係，在沒有 noise 的情況之下，它的 rank 只能有 3。

Slide 11 Factorization

所以我們能將 measurement matrix factorize 成 2m*3 和 3*n 的 matrix =>

know $\quad \underset{2m\times n}{\mathbf{W}} = \boxed{\underset{2m\times 3}{\mathbf{M}}\ \underset{3\times n}{\mathbf{S}}} \quad$ solve

則 M 代表的是 motion，S 代表 recover 的 shape。所以我們就 recover 出所有 camera 的 projection 的位置和所有 3D 中的點的位置。所以給定 W，我們想辦法分成 M 和 S，就是 Factorization。

因此利用 SVD 將 W 分成 $U\Sigma V^T$，Σ 是 n*n 對角矩陣，U 是 2m*n 的 orthogonal matrix，V 是 n*n orthogonal matrix。而 Σ 爲 sort 過後的對角矩陣，長相如下:

$$\Sigma = \begin{bmatrix} \lambda_1 & 0 & 0 & 0 \\ 0 & \lambda_2 & 0 & 0 \\ 0 & 0 & \lambda_3 & 0 \\ 0 & 0 & 0 & \ldots \end{bmatrix}$$

而因爲 W 的 rank 只有 3，所以理論上來說 λ_3 下面以後都是 0，而因爲有 noise 的關係，後面可能不爲 0。而 SVD 的好處就是告訴我們最靠近 W 而且 rank 爲 3 的 matrix。故將 λ_3 以後的值 assign 爲 0。因爲被 assign 爲 0，故只剩下前面三個 vector 比較重要，所以 Σ 變成 3*3 的 matrix，U 變成 2m*3 的 matrix，V 變成 3xn

$$\underset{2m\times n}{\mathbf{W}} = \underset{2m\times 3}{\mathbf{M'}}\ \underset{3\times n}{\mathbf{S'}}$$

的 matrix。而實際上我們需要的是兩個如右的 matrix:

而不是三個 matrix，故將Σ從中間切開，得到 M’ = $U\Sigma^{1/2}$、S’ =$\Sigma^{1/2}V^T$。因此我們就求出了最靠近 W 的 factorization，M’代表 projection matrix，S’代表 shape。

Slide 12 Metric constraints

Projection matrix 是從 rotation matrix 前面兩個 row 所提出來的。因此有幾個特性。首先 projection matrix 每一個 row 應該是一個 unit vector，而且每個 row 之間應該是要垂直的。所以說 projection matrix 乘上它自己的 transpose 應該是一個 identity matrix: $\prod\prod^T = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

而前面所求出的 M’並不一定滿足上面所述，故需要再乘上一個 transformation matrix A，才能得到滿足上述的 M。

$$\mathbf{M}'\mathbf{A} = \mathbf{M}$$

Trick (not in original Tomasi/Kanade paper, but in followup work)

- **Constraints are linear in AA^T :**

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \Pi\,\Pi^T = \Pi'\mathbf{A}\left(\mathbf{A}\,\Pi'\right)^T = \Pi'\mathbf{G}\,\Pi'^T \qquad \textit{where} \quad \mathbf{G} = \mathbf{A}\mathbf{A}^T$$

- **Solve for G first by writing equations for every $_i$ in M**

- **Then G = AA^T by SVD (since U = V)**

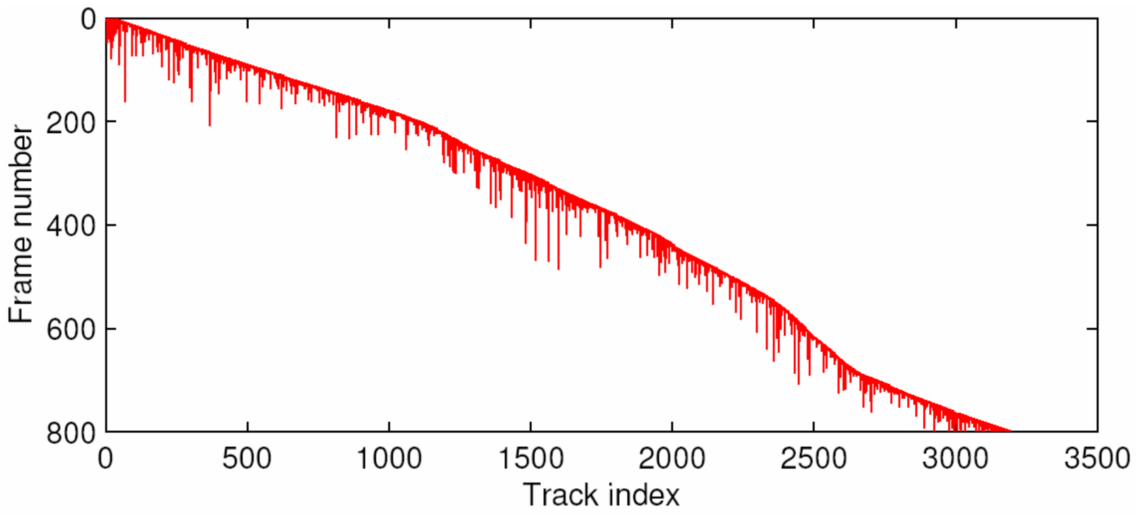
Slide 13 Factorization with noisy data

有 noise 的情況就是：$\underset{2m\times n}{\mathbf{W}} = \underset{2m\times 3}{\mathbf{M}}\ \underset{3\times n}{\mathbf{S}} + \underset{2m\times n}{\mathbf{E}}$

而用 SVD 的話就可以得到 rank 為 3 而最接近 W 的 approximation。而我們現在所做的是 assume orthogonal 的情況之下，因此在拍攝影片時，必須想辦法讓 focus 越遠越好，以免產生 perspective effect。做時由於沒考慮 missing data，因此 camera 不可以動得太遠，因此 camera 轉動時儘量保持中間有部份是重疊的，而利用中間重疊的部份幫助做 construction。

Slide 15 Why missing data?

之前假設 W 是 full 的，不過在做 tracking 的時候，我們可以發現每一個 tracking point 的生命期相對於一段 video 來說是相當短的，如下圖：



而理論上我們希望看到整片都是紅的，也就是 track point 能從第一個 frame 活到最後一個 frame。而實際上看到的是每一個 track point 最多只能活 100 個 frame，所以基本上就會有 missing data，也就是白色的部份。而爲什麼會有 missing date 呢? 第一個就是 occlusion，occlusion 包括 feature point 離開視角範圍。第二個就是 tracking failure，可能是 tracking 的 algorithm 或是程式寫得不夠好所造成。所以得到的 measurement matrix W 只有 partial field，沒辦法直接 apply factorization。

Slide 16 Tomasi & Kanade

Lucas Kanade 提出了 Hallucination/Propagation 的方法。來看的話，假設我們有四個點，四個 view，那我們的 measurement matrix 就是一個 8*4 的 matrix：

$$W = \left[\frac{U}{V}\right] = \begin{bmatrix} u_{11} & u_{12} & u_{13} & u_{14} \\ u_{21} & u_{22} & u_{23} & u_{24} \\ u_{31} & u_{32} & u_{33} & u_{34} \\ u_{41} & u_{42} & u_{43} & ? \\ v_{11} & v_{12} & v_{13} & v_{14} \\ v_{21} & v_{22} & v_{23} & v_{24} \\ v_{31} & v_{32} & v_{33} & v_{34} \\ v_{41} & v_{42} & v_{43} & ? \end{bmatrix}$$

其中第四個點在第四個 view 中沒有被看到，因此不可以直接 apply factorization matrix。不過理論上我們不需要這個第四個點就可以將這個 matrix

求出來。因爲原來的 matrix 所提供的是 redundant 的 information，所以 recover 的時候可以不需要這麼多點。因此我們可以將第四個 view 直接拿掉，相當於我們有三個 view，四個點，而我們就可以得到一個 6*4 的 full filled matrix：

$$W_{6\times 4} = \begin{bmatrix} u_{11} & u_{12} & u_{13} & u_{14} \\ u_{21} & u_{22} & u_{23} & u_{24} \\ u_{31} & u_{32} & u_{33} & u_{34} \\ v_{11} & v_{12} & v_{13} & v_{14} \\ v_{21} & v_{22} & v_{23} & v_{24} \\ v_{31} & v_{32} & v_{33} & v_{34} \end{bmatrix}$$

因此就可以直接 apply factorization 了。

Slide 17 Tomasi & Kanade

而求出來的就像: $W_{6\times 4} = R_{6\times 3} S + \mathbf{t}_{6\times 1} \mathbf{e}_4^T$

R: rotation matrix； t: translation matrix

而 4 個點的 shape 我們都知道，可是我們只知道 3 個 image 的 projection matrix，就是 rotation 加上 translation。所以理論上我們把有 missing point 的那張 image 整張不看,用剩下的資訊去 fit 出 shape 來,因爲這四個點在前面 3 張 image 都是被看到的，因此我們現在缺的是第四張 image 的 projection matrix。

Slide 18 Tomasi & Kanade

我們現在要求的就是 i_4 和 j_4，而前面 3 個點的 shape 已知，且已知 3 個點落再第四張 image 中的位置，因此我們就可以列出：

$$\begin{bmatrix} u'_{41} & u'_{42} & u'_{43} \end{bmatrix} = \mathbf{i}_4^T \begin{bmatrix} \mathbf{s}'_1 & \mathbf{s}'_2 & \mathbf{s}'_3 \end{bmatrix}$$
$$\begin{bmatrix} v'_{41} & v'_{42} & v'_{43} \end{bmatrix} = \mathbf{j}_4^T \begin{bmatrix} \mathbf{s}'_1 & \mathbf{s}'_2 & \mathbf{s}'_3 \end{bmatrix}$$

來反求 i_4 和 j_4。不過因爲我們假設的是 orthogonal，因此只有考慮 rotation matrix 中的 i_4 和 j_4。而下面這些式子所做的事，只是 shift 的動作，使 3D 和 2D 的原點是在 centroid。

$$u'_{4p} = u_{4p} - a'_4 \qquad v'_{4p} = v_{4p} - b'_4 \qquad \mathbf{s}'_p = \mathbf{s}_p - \mathbf{c}$$

$$a'_4 = \frac{1}{3}(u_{41} + u_{42} + u_{43}) \qquad b'_4 = \frac{1}{3}(v_{41} + v_{42} + v_{43}) \qquad \mathbf{c} = \frac{1}{3}(\mathbf{s}_1 + \mathbf{s}_2 + \mathbf{s}_3)$$

將這些 shift 過後的點代入線性系統中來求解。

$\begin{bmatrix} u'_{41} & u'_{42} & u'_{43} \end{bmatrix}$ 為 2*3 的 matrix，$\begin{bmatrix} \mathbf{s}'_1 & \mathbf{s}'_2 & \mathbf{s}'_3 \end{bmatrix}$ 為 3*3 的 matrix，所以 i_4 為 2*3 的 matrix，因此我們要求的解有六個變數。算出來的 i_4 和 j_4 就可以將之補回去成為 R_{8*3}，也就是 4 個 image 的 orthogonal rotation matrix。

Slide 19 Tomasi & Kanade

剛剛作法是把 row 拿掉。理論上，也可以將 column 拿掉。拿掉 missing data 的 column 後，我們就有 4 個 image，3 個 point，然後就可以算出 4 個 image 的 projection matrix，算出後，再想辦法將第 4 個 3D 的點加進來。

$$W_{8\times 3} = \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ u_{21} & u_{22} & u_{23} \\ u_{31} & u_{32} & u_{33} \\ u_{41} & u_{42} & u_{43} \\ v_{11} & v_{12} & v_{13} \\ v_{21} & v_{22} & v_{23} \\ v_{31} & v_{32} & v_{33} \\ v_{41} & v_{42} & v_{43} \end{bmatrix}$$

Slide 20 Tomasi & Kanade

這個方法最大的問題在於，必須要找出最大的 full submatrix，而這個問題基本上是很困難的問題，而且是 NP-hard 的問題。並且資料不像 factorization 一樣，所有的點同時丟進去，大家的地位是一樣的，而是利用前面所算出來的結果來估計我現在的值，這樣就會有誤差，而這個 error 會 propagate。

Slide 21 Shum, Ikeuchi & Reddy

因爲前述的缺點，因此有人提出較好的 factorization with missing data point 的方法。Shum, Ikeuchi 和 Reddy 在 1995 年提出了 PCA 的 factorization 的方法。Shum 現在是 Microsoft Research 的一個 director；Ikeuchi 以前是 CMU 的教授，現在在日本東京大學做 Computer Vision；而 Reddy 是 1995 年的 Turing Award 的得主。這個方法是 Shum 在 CMU 唸 Ph. D.時的一個論文。

而 PCA 和 SVD 爲什麼很相似呢? 是因爲每一個 matrix W 基本上有 n 個 m-d 的 vector，而 PCA 基本上就是算它們的 mean vector 和 covariance sigma。而基本上只要就是要找到這種關係：$\left\|\mathbf{W}-\mathbf{e}\mathbf{t}^{\mathrm{T}}-\mathbf{U}\mathbf{S}\mathbf{V}^{\mathrm{T}}\right\|$

使它們相減的 norm 最小。理論上沒有 missing data 的話結果是一樣的，但是如果有 missing data 的話其實會比較好處理。因爲我們要找 USV^T 和 t 使得這個 norm 最小，相對來說就是對於 W 裡面每一個 q_{ij} 使其 entry 的 norm 最小。而有 missing data 的話，那個 entry 就可以不用考慮了。因此我們將 SVD 的問題轉成 least square 問題。而因爲我們得 W 的 rank 是 r，因此 USV^T，其分別爲 U 爲 m*r 的,matrix；S 爲 r*r 的 matrix；V^T 爲 r*n 的 matrix。因此將 SVD cast 成 least square 的問題，看的到的 entry 才接受它的 error contribution，看不道的就不理它了。因此目標是要找最好的 USV^T 最靠近 W，並且 rank 爲 r，因此是對 uv 做 minimization，找一組 uv 使得Φ爲最小。

$$\min \phi = \frac{1}{2}\sum_{q_{ij}\text{ is visible}}\left(\mathbf{q}_{ij}-\mathbf{t}_j-\mathbf{u}_i^{\mathrm{T}}\mathbf{v}_j\right)^2$$

Slide 22 Shum, Ikeuchi & Reddy

假設 W 爲 m*n 的 matrix；S 爲 r*r 的 matrix，而其 degree of freedom 爲 r；V 爲 r*n 的 matrix，degree of freedom 爲 n*r-n-C_2^n；U 爲 m*r 的 matrix，degree of freedom 爲 m*r-m-C_2^m。而 U 爲 m*r 的 matrix，所以理論上會有 m*r 個變數，但是 U 爲 orthogonal 的 matrix，因此每一個 column 的 norm 爲 1，又任兩個 column 之間是垂直的，因此 degree of freedom 才會是 m*r-m-C_2^m。V 的情況同理。而整個 USV^T 的 degree of freedom 爲 dofTotal = dofU+dofS+dofV，因此當 W 裡面只要有超過 dofTotal 個 entry 有值，就可以 recover 回來，因此比 defTotal 多的話，我們就求 least square 的解；相反的話，不夠的話就直接 return。

Slide 23 Shum, Ikeuchi & Reddy

因為 u 和 v 都是變數，故此問題為 non-linear 的問題。故首先固定 v，變成 linear function 就可以解 u；求出 u 之後，固定 u，再求 v。反復去求解直到收斂為止。Algorithm 如下：

1) initialize v
2) update $\mathbf{u} = \mathbf{B}^{+}(\mathbf{w} - \mathbf{t})$
3) update $\mathbf{v} = \mathbf{G}^{+}\mathbf{w}$
4) stop if convergence, or go back to step 2

而 initial 的 v 在 paper 之中是 randomly 產生出來的，可是通常 randomly 產生的 v 結果不會太好，有興趣的話，詳細部份可以去看 paper。

Slide 24 Shum, Ikeuchi & Reddy

這個方法的缺點就是對於一開始 initial 的值非常 sensitive，所以如何取一開始的點就很重要了。

Slide 25 Linear fitting

再來介紹如何 initialize 這個 v。Linear fitting 這個方法本來就是一個用來解決 factorization with missing data 的方法，問題它不是 iterative，它不保證你會得到 local optimal。所以我們可以用這個解出來的解，來做我們前面 procedure 所要 initialize 的那個 v，使得我們確保我們會到 local optimal 上面去。而這個方法是 Jacobs 在 2001 年所提出來來的 linear fitting 的方法。

我們試著去找出 rank 為 r 的 $\widehat{W}$，而且越靠近 W 越好： $\left\|\hat{\mathbf{W}} - \mathbf{W}\right\|$

假設 W 是一個 m*m 的 matrix，而每一個 column 代表一個 vector，所以 W 是 n 個 m-d 所形成的 vector space，所以理論上 W 可以是一個 m dimension 的 vector space，可是我們知道 W 的 rank 為 3，因此這些 vector 之間會有一些線性相依的關係，所以這個 vector space 實際上為 dimension 為 3 的 vector space。

假設我們有個 matrix A 為 3*3，而實際上，上面的的點都是從 X+2Y+Z=0 這個平面上取得。因此理論上 A 若是 full rank 會 span 一個三維的空間，可是因為每個 column 之間有線性相依的關係，因此這個 case 裡的 space 實際上只有二維。

所謂的 SVD 基本上就是說，給你一個 matrix，它去找出它真正所對應的 vector space 在哪裡。

Slide26 Linear fitting

所以如果這個 case 有 noise 的話，我們的 matrix 爲 3*n，而 rank 實際上爲 2 的話，意思就是說 SVD 要找一個平面，使得這 n 個點投影到這個 3D 中的平面的 distance 要最小。SVD 基本上就是在做這件事。然後將算出來的投影點填回去這個 matrix，其 rank 就會是 2。

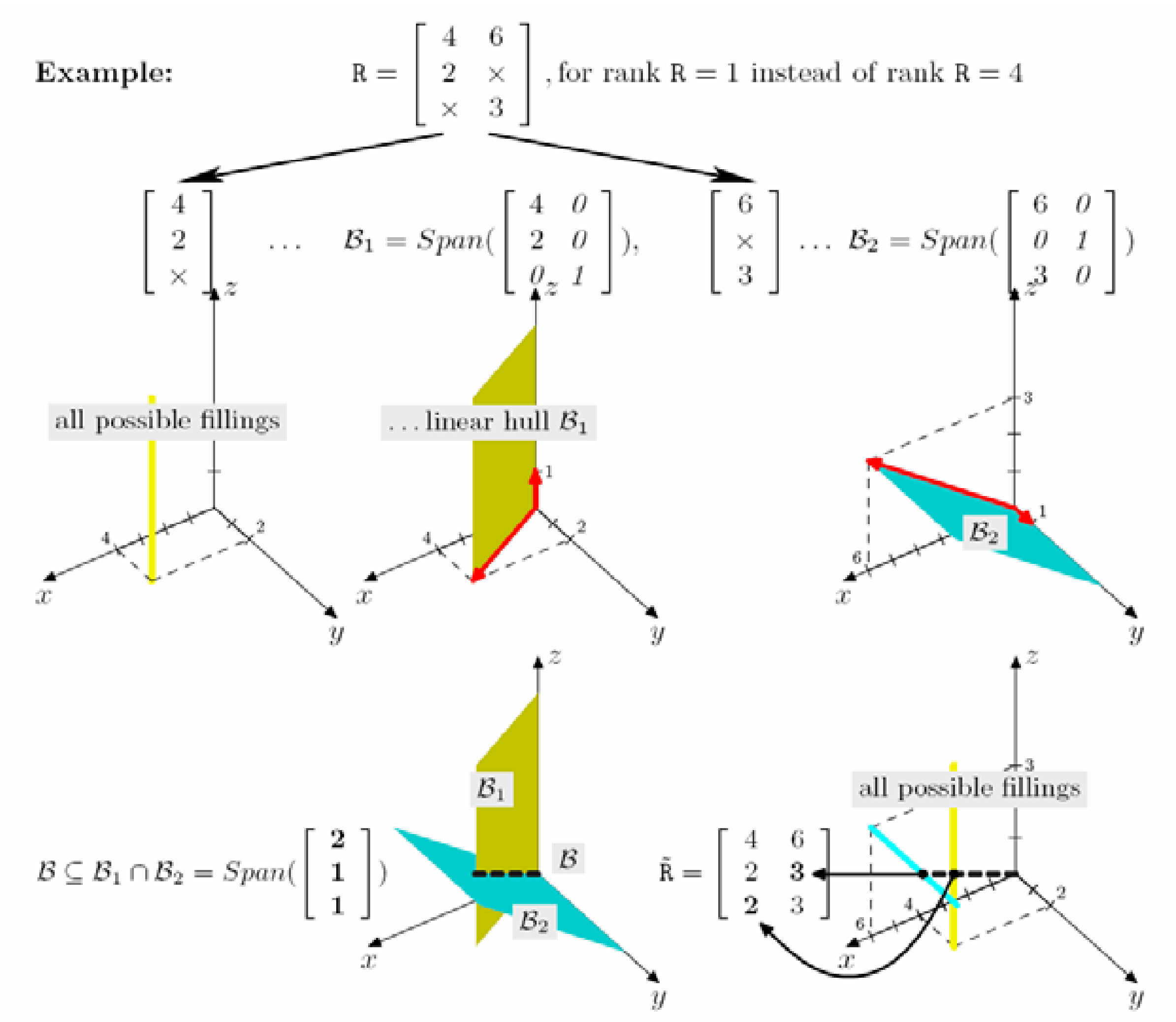
所以如果沒有 noise 的情況下，你的 M 的這個 matrix 的 rank 是 3 的話，相當於在 M 的 column 任取三個 vector，他們就會決定你的 vector space 是什麼。可是現在問題在於找出我們有 missing data，所以我們任取了三個 column 所形成的 space，L 一定是它的子集合。

$$\mathrm{L} \subseteq \bigcap_{(i,j,k)} span(A_i, A_j, A_k)$$

所以我們可以取多個 column vector 所形成的 space，然後我們的 L 就是在它們的交集裡面，不過任取出的 column space 不可以有線性相依的關係。

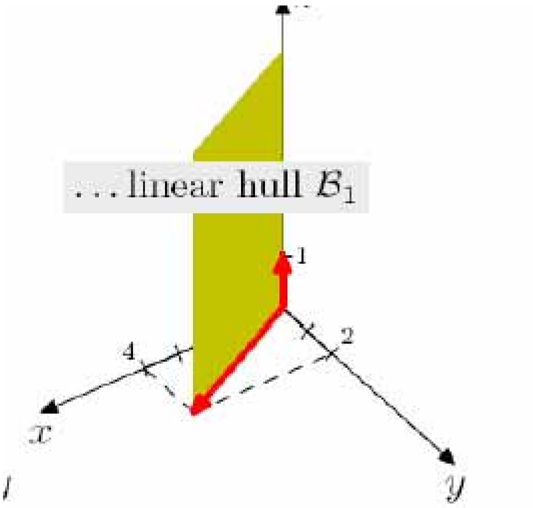
Slide 27

在這個 example 裡面，我們要找出 rank 爲 1 的 matrix 使其最靠近 R。

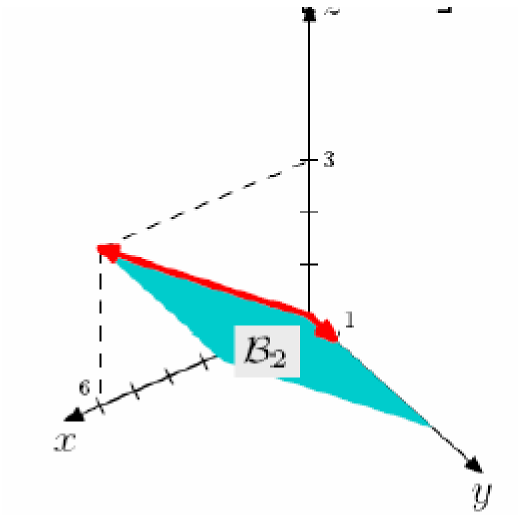


因爲 rank 爲 1，我們一次取一個 column vector，使其成一個 vector space，然後將這些 vector span 做交集，所得到的 vector space 就是我們要求的 vector space。所以現在我們有兩個 1D 的 vector [4, 2, X]和[6, X, 3]。

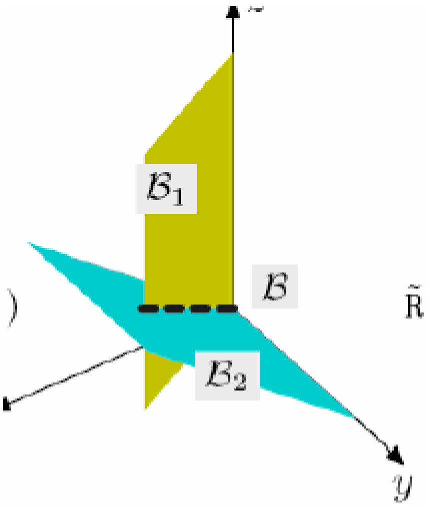
而[4, 2, X]所形成的 vector space 如下:



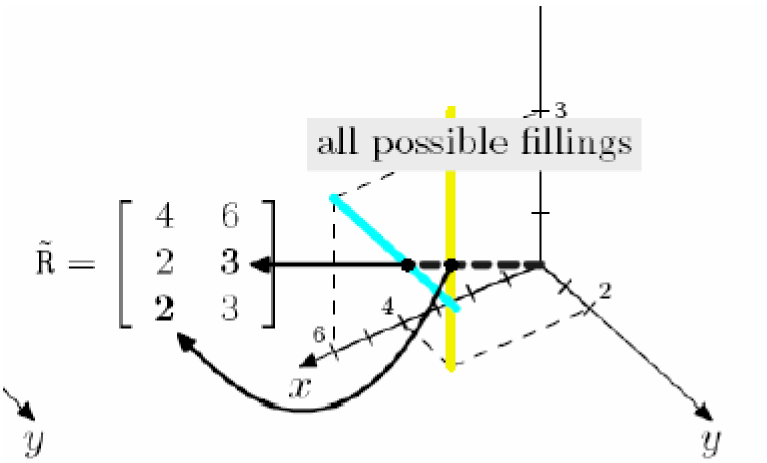
而[6, X, 3]所形成的 vector space 如下:



然後兩個 vector space 的交集就是這條虛線:



這條虛線所對應的 vector space 就是 rank 爲 1，且靠近 R 的那一個 matrix 所形成的 vector space，也就是我們要求的 approximation matrix。所以我們就可以求出:



故由此 2D 的例子，可以 apply 到 3D 中。但若是所求出的 L 要的 rank 爲 3，可是求得的確是 4 的話，代表 constraint 不足。

Slide 28 Linear fitting

我們要求的那個 vector space 就是將所有 triplet 所形成的 vector space 的交集，包含或等於這個 vector space。問題是這東西對 matrix 來講很難算。所以我們用 DeMorgan's Raw 先找出$\bar{L}$，也就是 L 的 complement space，基本上就是所有 triplet 所形成的 vector space 的所有的 complement space 的聯集，一但求出來就可以求出 L。

$$\mathrm{L} = \bigcap_t S_t \Rightarrow \bar{\mathrm{L}} = \bigcup_t \bar{S}_t$$

所以每一個$\bar{S}_t$可用 Nt 來表示，Nt 是一個 matrix，這個 matrix 描述的是一個 vector space，這個 vector space 代表的就是$\bar{S}_t$。意思就是這個 matrix 裡面的每一個 column vector 都垂直於 St 這個 space。而 Nt 代表的就是$\bar{S}_t$這個 vector space。而聯集呢，就是將這群 Nt 全部綁在一起，變成一個很大的 vector space => $N=[N_1, N_2, \ldots N_l]$

而我們要求的是 L，因此要求的就是 N 的 complement space，也就是 null space。因此利用 SVD，找到最小的 3 個 λ 所對應的那三個 vector 所形成的 space 就是 L 這個 space。而如果第小的 λ 其 singular value 太小，小於 0.001 的話，意思就是說這其實不是一個很好的 approximation，而這個 null space 的 rank 不是 3 而是 4，因此就不是我們要的 vector space，因此就 return 結果 unstable。

這個方法可以直接 apply 用來解 singular value with missing data，或者是用它求出來的結果去 initialize 我們剛剛所提到的 PCA m-d 的方法，去求出更精確的結果。這個部份可以看 paper，或是 matlab 的 code 來 figure out 出他們的方法。

Slide 30 Factorization for projective projection

3D 空間中有 n 個點，將每個點分別乘上 m 個 projection matrix，即可算出每個點投影在 m 個 image 上的位置。其中 λ 爲 projection 的深度值。

如果 λ 已知，則可用 rank 爲四的 factorization，但 λ 僅能由估計得知。

第 i 個是 p_i 乘上一個 projection matrix (3*4)，$\prod_j$ 。把第 i 個點投影到第 j 個 image 上面得到一個 q_{ij} 就是 image coordinate，而前面還有一個 λ_{ij} 就是 projective depth。

projective depth

$$\lambda_{ij}\mathbf{q}_{ij} = \Pi_j \mathbf{p}_i$$

對一個點的情況如上所示。而每一個點在每一個 image 的座標堆起來變成大 matrix 的話就可以表達成這樣：

$$\underbrace{\begin{bmatrix} \lambda_{11}\mathbf{q}_{11} & \lambda_{11}\mathbf{q}_{12} & \cdots & \lambda_{11}\mathbf{q}_{1n} \\ \lambda_{21}\mathbf{q}_{21} & \lambda_{22}\mathbf{q}_{22} & \cdots & \lambda_{2n}\mathbf{q}_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \lambda_{m1}\mathbf{q}_{m1} & \lambda_{m2}\mathbf{q}_{m2} & \cdots & \lambda_{mn}\mathbf{q}_{mn} \end{bmatrix}}_{3\mathrm{m}\times\mathrm{n}} = \underbrace{\begin{bmatrix} \mathbf{\Pi}_1 \\ \mathbf{\Pi}_2 \\ \vdots \\ \mathbf{\Pi}_m \end{bmatrix}}_{3\mathrm{m}\times 4} \underbrace{\begin{bmatrix} \mathbf{p}_1 & \mathbf{p}_2 & \cdots & \mathbf{p}_n \end{bmatrix}}_{4\times\mathrm{n}}$$

Slide 31 Sturm & Triggs

$$\lambda_{ip} = \frac{(\boldsymbol{e}_{ij} \wedge \boldsymbol{q}_{ip}) \cdot (\boldsymbol{F}_{ij}\boldsymbol{q}_{jp})}{||\boldsymbol{e}_{ij} \wedge \boldsymbol{q}_{ip}||^2} \lambda_{jp}$$

同一個 3D 空間中的點 p 在任兩個 image plane 上的投影的深度值有 coherency 的關係，如上公式所示。其中 eij 爲第 i 個 image 及第 j 個 image 上的 epipole，Fij 爲兩個 image 之間的 fundamental matrix。所以若已知一個點 p 投影到 image j 的深度值，即可利用以上的公式求得同一個點投影到 image i 的深度值。演算法有八個步驟：

(1) 將每一張 image i 的 cordinates 利用 Ti 先作 normalize。
(2) 估計 image 兩兩之間的 fundamental matrix 及 epipoles。
(3) Initialize λ_{ip} 爲 1，利用上面的式子求出其他所有的 λ 值。
(4) 利用求出的 λ 值，算出 matrix W。
(5) 將 matrix W 作 balance。
(6) 將 matrix W 作 SVD 分解。
(7) 利用 SVD，重建 projective motion 及 shape。
(8) 利用 T_i，將重建到的 motion 回復成 step 1 作 normalize 之前的 projective motion。

Slide 32 Sturm & Triggs

Compute an initial estimate of the projective depths z_{ij}, with $i = 1, .., m$ and $j = 1, .., n$.
Repeat:
(1) normalize each row of the data matrix $\mathcal{I}$, then normalize each one of its columns;
(2) use singular value decomposition to compute the matrices $\mathcal{M}$ and $\mathcal{P}$ minimizing $|\mathcal{I} - \mathcal{MP}|^2$;
(3) for $i = 1, .., m$ and $j = 1, .., n$, find the value of z_{ij} minimizing $|z_{ij}\boldsymbol{p}_{ij} - \mathcal{M}_i\boldsymbol{P}_j|^2$ using linear least squares;
until convergence.

之前的方法只有作一次 iteration 即停止，但若多作幾個 iteration 可讓估計出來的 M 和 P 更爲正確。即每次作完 SVD 後，即固定 M 和 P 然後調整深度值，之後再固定深度值，重新作一次 SVD，算出新的 M 和 P，重複幾個 iteration，即可得到較佳的 M 和 P。

Slide 33 Mahamud *et. al.*

Compute an initial estimate of the projective depths z_{ij}, with $i = 1, .., m$ and $j = 1, .., n$ and normalize each column of the data matrix $\mathcal{I}$.
Repeat:
(1) use singular value decomposition to compute the matrices $\mathcal{M}$ and $\mathcal{P}$ minimizing $|\mathcal{I} - \mathcal{MP}|^2$;
(2) for $j = 1$ to n, compute the matrices $\mathcal{R}_j$ and $\mathcal{Q}_j$ and find the value of $\boldsymbol{z}_j$ maximizing $|\mathcal{R}_j\boldsymbol{z}_j|^2$ under the constraint $|\mathcal{Q}_j\boldsymbol{z}_j|^2 = 1$;
(3) update the value of $\mathcal{I}$ accordingly;
until convergence.

Projection matrix 而且有 missing data：用 Mahamud 的方法(只講概念)
基本上跟前面的概念一樣，找一個 matrix 使得它跟 measurement，而且它的 rank 必須爲 4。

因爲我們希望能找到 z、M、P 可以解釋 projection points，所以要去找 z、M、P 去 minimize 這個 energy function。

Slide 36 Mahamud *et. al.*

這頁投影片主要是在說明 Energy function $E_{ij} = \sum_{ij} | z_{ij} q_{ij} - M_i P_j |^2 = \sum_{ij} | q_{ij} \times (M_i P_j) |^2$

而 $E_i^{(P)} = \sum_{j=1}^{n} \left| q_{ij} \times (M_i P_j) \right|^2 = \left| C_i m_i \right|^2$

其中 C_j =

$$\begin{pmatrix} -w_{i1}\boldsymbol{P}_1^T & \mathbf{0}^T & u_{i1}\boldsymbol{P}_1^T \\ \mathbf{0}^T & -w_{i1}\boldsymbol{P}_1^T & v_{i1}\boldsymbol{P}_1^T \\ -v_{i1}\boldsymbol{P}_1^T & u_{i1}\boldsymbol{P}_1^T & \mathbf{0}^T \\ \cdots & \cdots & \cdots \\ -w_{in}\boldsymbol{P}_n^T & \mathbf{0}^T & u_{in}\boldsymbol{P}_n^T \\ \mathbf{0}^T & -w_{in}\boldsymbol{P}_n^T & v_{in}\boldsymbol{P}_n^T \\ -v_{in}\boldsymbol{P}_n^T & u_{in}\boldsymbol{P}_n^T & \mathbf{0}^T \end{pmatrix}$$

Slide 37 Mahamud *et. al.*

Compute an initial estimate of the vectors $\boldsymbol{P}_1 \ldots, \boldsymbol{P}_n$ and normalize these vectors.
Repeat:
(1) for $i = 1$ to m, compute the unit vector $\boldsymbol{m}_i$ that minimizes $|\mathcal{C}_i \boldsymbol{m}_i|^2$;
(2) for $j = 1$ to n, compute the unit vector $\boldsymbol{P}_j$ that minimizes $|\mathcal{D}_j \boldsymbol{P}_j|^2$;
until convergence.

(不用把所有的 data 都算進來，有 data 的時候才算，所以可以處理有 missing data 的問題)。

簡單的來說，這個演算法就是，先 initialize P_{ij}，然後固定 P 去求可以 minimize $| c_i m_i |$的 m_i，然後固定 m 去求可以 minimize $| D_j P_j |$ 的 P_j，一直 iterate 到它 convergence 爲止。