30/10

we have seen it for 2 images...
it is possible to do it with many images

Image Mosaicking via Synchronization

Exercise + Theory

Image Analysis and Computer Vision

Federica Arrigoni – federica.arrigoni@polimi.it

POLITECNICO
MILANO 1863

Exercise Session – October 30, 2024

→ theory introduction for multi image mosaiking

Outline

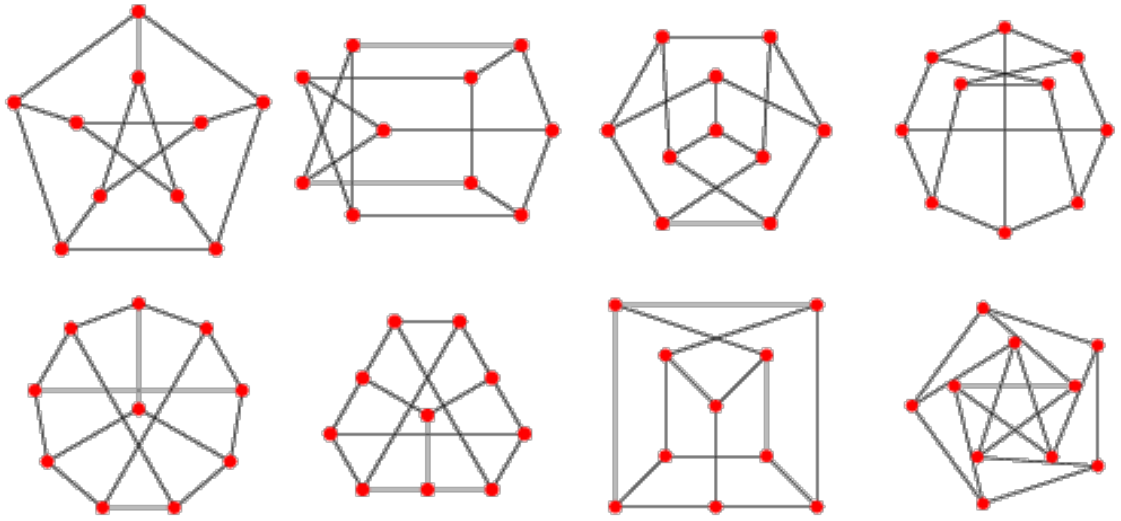
- Background
- Introduction → to synchronization problem... tool for image mosaiking
- Spectral solution ⟷ provide a solution to that! general solution
- Application to image mosaicking → how to use it
- Conclusion & extensions → extend methodology for other problems
- Matlab demo examples

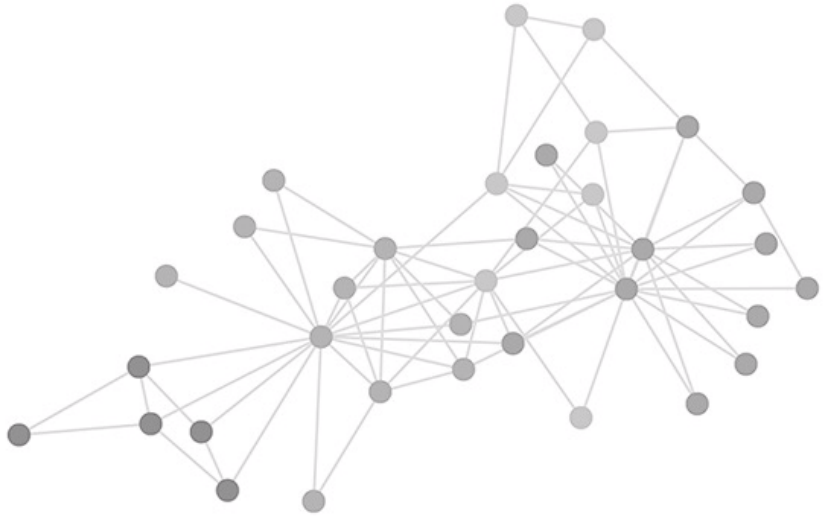
Outline

- **Background**
- Introduction
- Spectral solution
- Application to image mosaicking
- Conclusion & extensions
- Matlab demo

Elements of Graph Theory

Definition. *A graph is a pair (V,E) where where V is a set whose elements are called* **vertices/nodes** *and E is a set of paired vertices, which are called* **edges**.





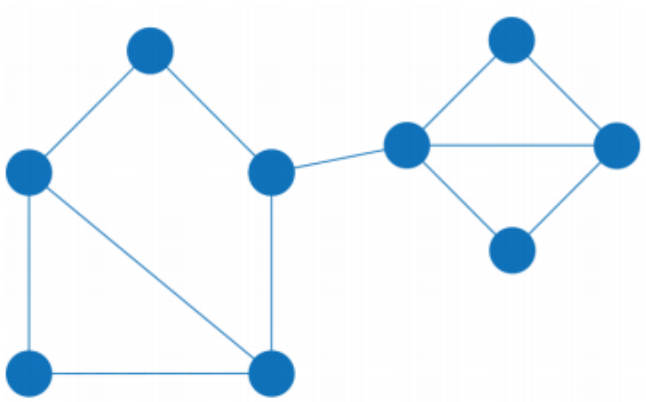
Notation: n = #V, m=#E

we will use GRAPHS to model relationships between images → each Node ≡ image ⇒

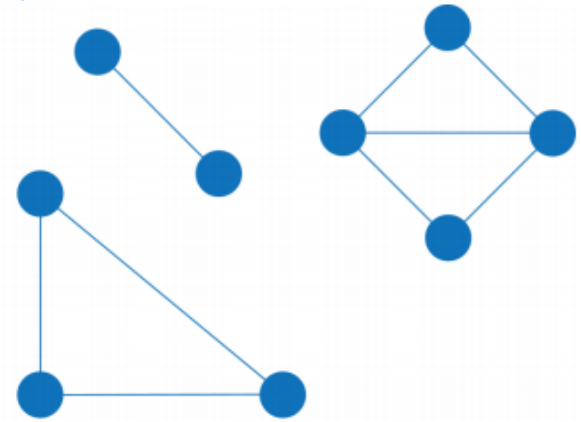
Elements of Graph Theory

Definition. *A graph is called* **connected** *if there exists a path from each vertex to any other.*

↓ We will assume Graphs to model images and their relationship are connected!



Connected Graph ✅

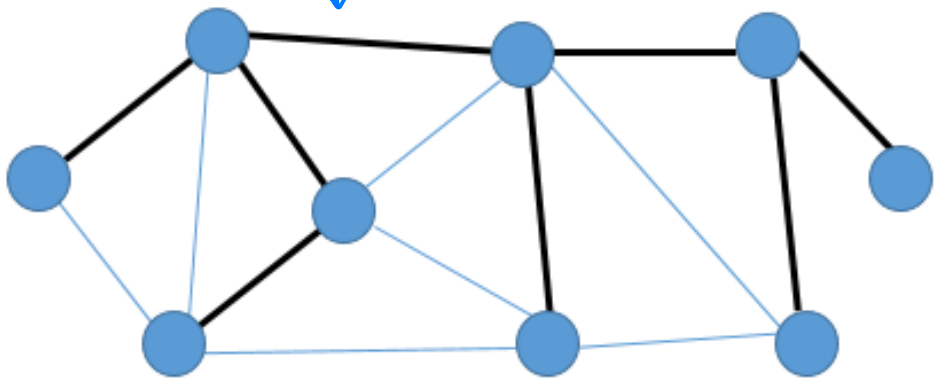


Disconnected Graph ❌
Includes 3 components.

Elements of Graph Theory

Definition. *A* **spanning tree** *in a graph is a minimal set of edges that connect all vertices.*

minimal set, reach all vertices



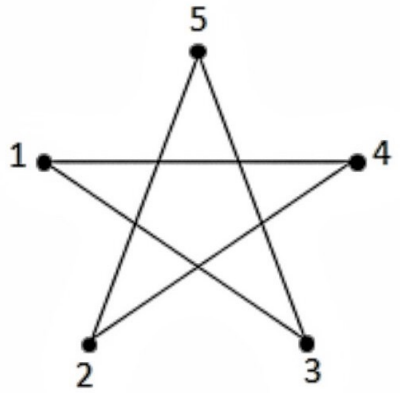
we will use 2 matrix

Elements of Graph Theory

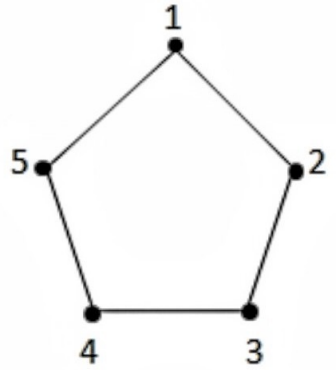
Definition. *The* **adjacency matrix** *A of a graph (V,E) with n nodes and m edges has dimension n x n and it is constructed as follows:*

$$[A]_{i,j} = \begin{cases} 1 & \text{if } (i,j) \in E \\ 0 & \text{otherwise} \end{cases}$$

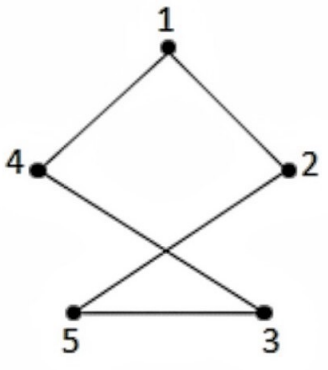
binary matrix encoding nodes relationship



	1	2	3	4	5
1			1	1	
2				1	1
3	1				1
4	1	1			
5		1	1		



	1	2	3	4	5
1		1			1
2	1		1		
3		1		1	
4			1		1
5	1			1	



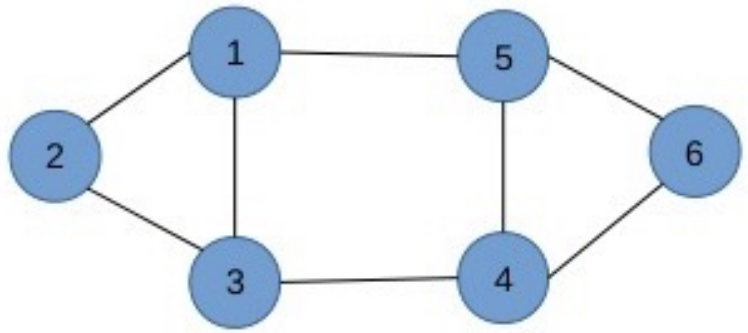
	1	2	3	4	5
1		1		1	
2	1				1
3				1	1
4	1		1		
5		1	1		

Elements of Graph Theory

Definition. *The **degree** matrix D of a graph (V,E) with n nodes and m edges is a diagonal matrix with dimension n x n such that:*

$$[D]_{i,i} = \#\{\text{edges connected to node } i\}$$

each term store the degree of a node



$$D = \begin{bmatrix} 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 \end{bmatrix}$$

second vertex has deg = 2, so it has 2 connecting nodes

↓

useful to count # relationship ∀ node

Kronecker product

NOT restrictions on dimensions (general)

Definition. Let A and B be two matrices of dimension m × r and n × s respectively. The Kronecker product of A and B is a mn x rs matrix denoted by $A \otimes B$:

$$A \otimes B = \begin{bmatrix} [A]_{1,1}B & [A]_{1,2}B & \dots & [A]_{1,r}B \\ [A]_{2,1}B & [A]_{2,2}B & \dots & [A]_{2,r}B \\ \dots & & & \dots \\ [A]_{m,1}B & [A]_{m,2}B & \dots & [A]_{m,r}B \end{bmatrix}$$

m×r

m×s

rs

m·m

take each entry of A and multiply by entire B matrix

general formula

Example:

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \otimes \begin{bmatrix} 7 & 5 \\ 6 & 0 \end{bmatrix} = \begin{bmatrix} 1\begin{bmatrix} 7 & 5 \\ 6 & 0 \end{bmatrix} & 2\begin{bmatrix} 7 & 5 \\ 6 & 0 \end{bmatrix} \\ 3\begin{bmatrix} 7 & 5 \\ 6 & 0 \end{bmatrix} & 4\begin{bmatrix} 7 & 5 \\ 6 & 0 \end{bmatrix} \end{bmatrix} = \begin{bmatrix} 7 & 5 & 14 & 10 \\ 6 & 0 & 12 & 0 \\ 21 & 15 & 28 & 20 \\ 18 & 0 & 24 & 0 \end{bmatrix}$$

create bigger matrix

Outline

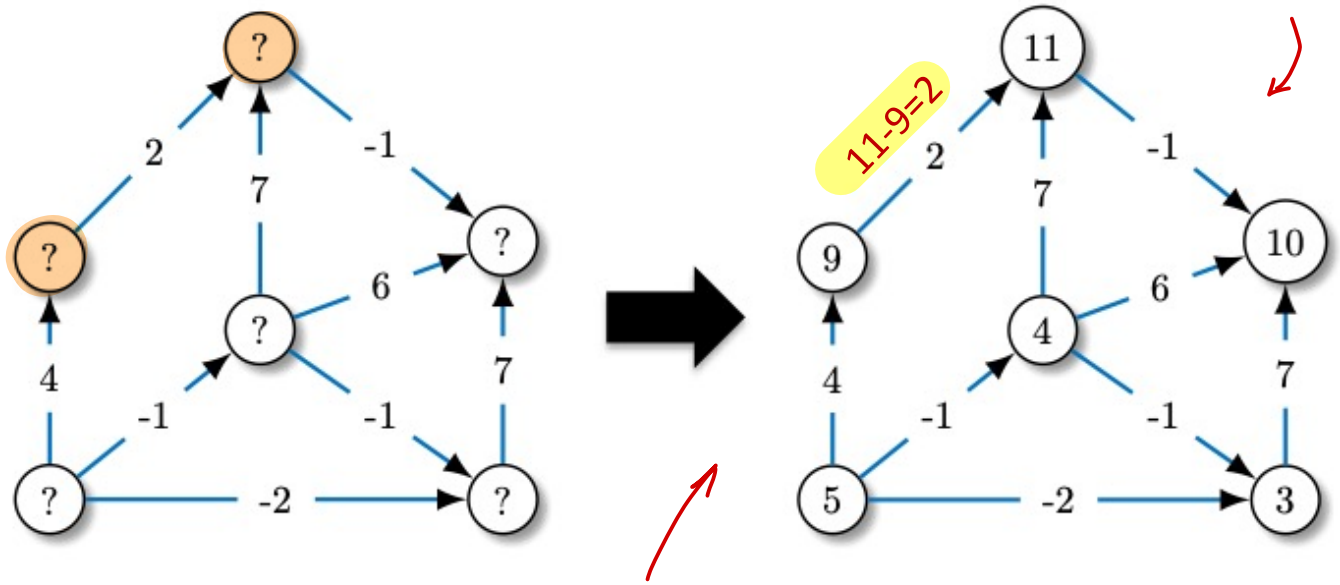
- Background
- **Introduction**
- Spectral solution
- Application to image mosaicking
- Conclusion & extensions
- Matlab demo

[INTRODUCTION of synchronization metodology, used to solve MOSAICING of image]

Example (intuition of problem)

⇓

Consider a network of nodes where each node is characterized by an unknown state, and pairs of nodes can measure the **difference** between their states.



↘ we know the difference between these values on the edges

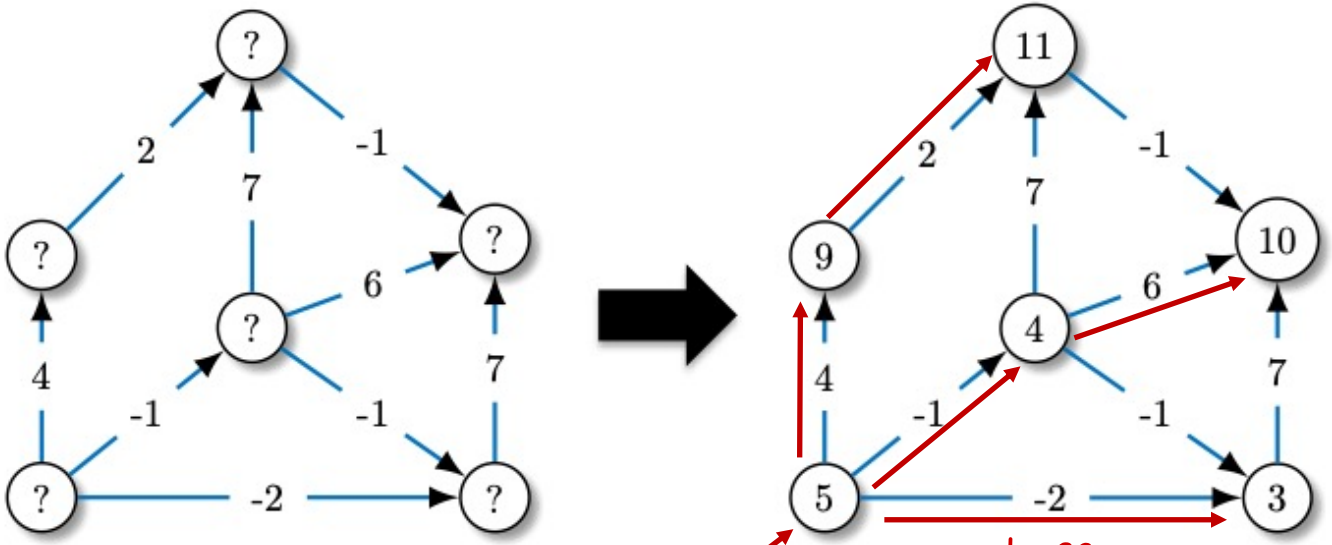
The goal is to infer the unknown states from the pairwise measures.

↗ find values

{ this satisfy constraint on the edge } ← possible solution ≡ assigning nodes values

Example

Consider a network of nodes where each node is characterized by an unknown state, and pairs of nodes can measure the **difference** between their states.



as we can OBSERVE..

HINT on how this solution has been constructed...

We chose a node as tree root, then start from there and sequentially propagate constrate

Using the notion of TREE

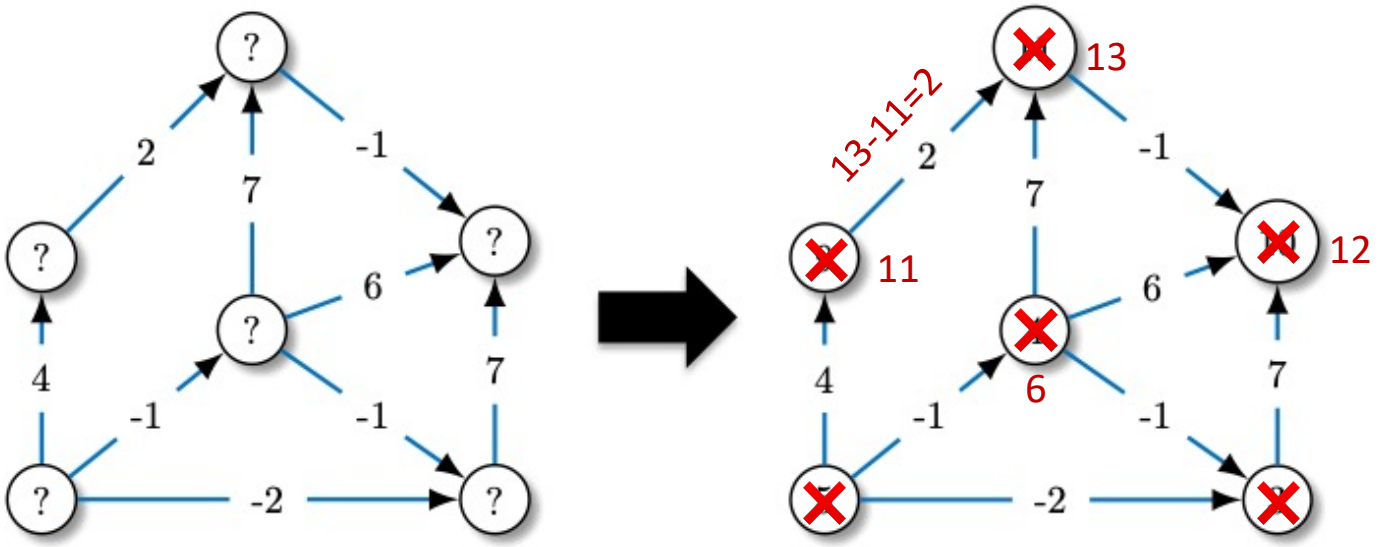
👉 A solution can be obtained by **sequential propagation**, starting from any node. The value of the initial node is **arbitrary**.

and inverse can allow to propagate

Example

Consider a network of nodes where each node is characterized by an unknown state, and pairs of nodes can measure the **difference** between their states.

NOT UNIQUE ↑ add SAME value to all the node ↑



We can get many solutions ↑

👉 The solution is **not unique**: adding the same constant to all the nodes gives another valid solution.

→ choice of initial node and value is arbitrary

NOT UNIQUE also because is solution up to a constant...

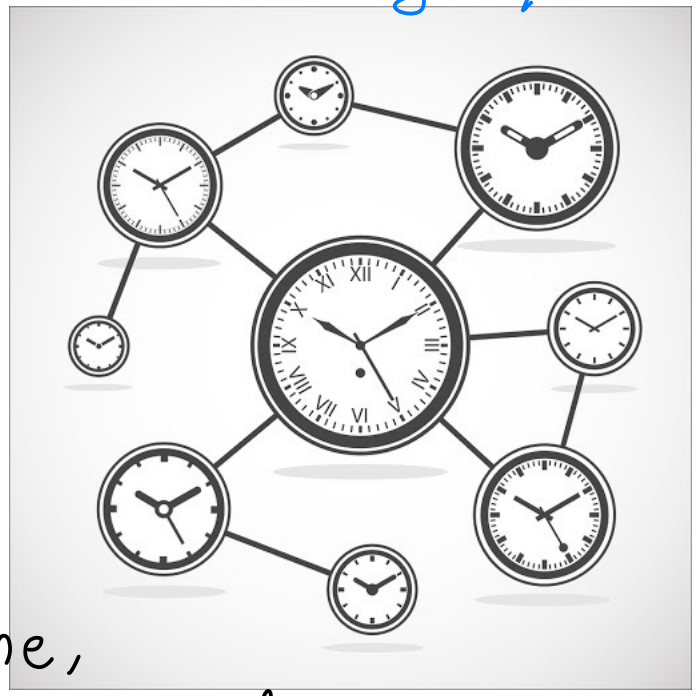
this PROBLEM can be generalized to approach image mosaicing

Example

COMMENT...
Why this name

extension of the PROBLEM of synchronizing clocks

Clock/time synchronization



Consider a GRAPH of clocks

- Each node has its own clock that tells the **time**.
- Each edge represents the ability to transmit and receive messages between the corresponding pair of nodes, which enables to measure clock **differences**.

Graph with many clocks, we don't know each time, but the pair values... task → synchronize all in unique reference system

The task is to enable each node in the network to convert its own time to that of a **unique common clock** (e.g. the clock of node 1).

A. Giridhar, P. Kumar. *Distributed clock synchronization over wireless networks: algorithms and analysis*. Conference on Decision and Control (2006)

Generalization

"group": there are many groups in CV, for example ROTATIONS, etc transformation which has operation

Problem Definition

The problem of clock synchronization can be generalized to the case where states are elements of a **group**. (instead of having numbers...)

Definition. *A group is a set Σ equipped with a binary operation $*$ that combines any two elements to form a third element, such that:*

the third element is still of Σ group

- **Associativity:** *for all $x, y, z \in \Sigma$ we have $(x * y) * z = x * (y * z)$.*
- **Identity:** *there exists an element 1_Σ such that for every $x \in \Sigma$ we have $1_\Sigma * x = x = x * 1_\Sigma$*
- **Inverse:** *for each $x \in \Sigma$ there exists an element $u \in \Sigma$ such that $x * u = 1_\Sigma = u * x$. Such element is commonly denoted by x^{-1}.*

(SATISFY PROPERTY!)

Clock synchronization: integer (real) numbers

Examples in Computer Vision: rotations, permutations, …

ex: homography respect this GROUP property!

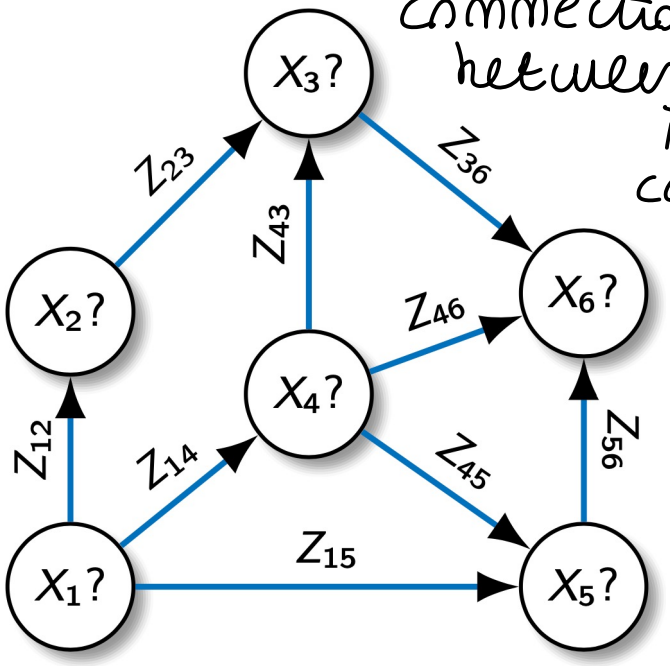
↳ in CV many problems use this formalism ...

Now general operation, can be MATRIX ↑ multiplication

Problem Definition

⇓ Generalize concept... before we consider difference

Graph representation: Nodes = unknowns, Edges = observations



connection between unkn is constrained

Time synchronization

$$Z_{ij} = X_i - X_j = X_i + (-X_j)$$

trivial extension

General case (group)

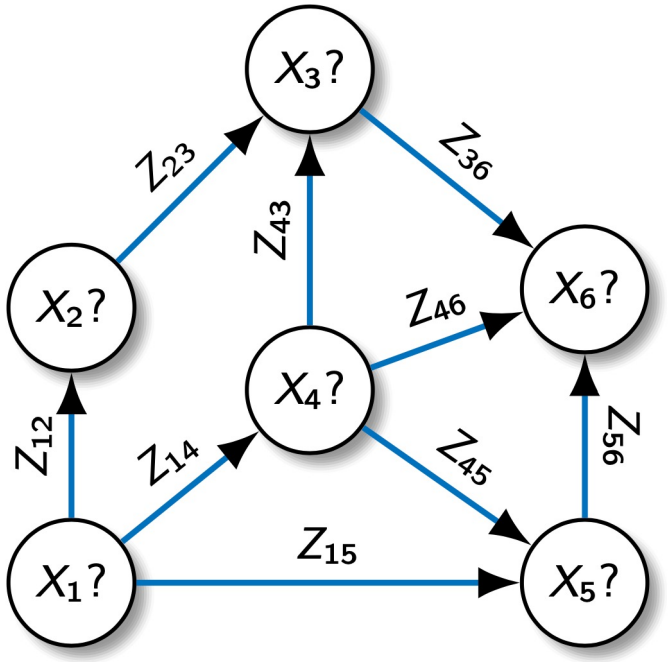
$$Z_{ij} = X_i * X_j^{-1}$$

inverse (like in MATRIX inversion)

Problem Definition

for example MATRICES

The goal (task) of synchronization is to recover elements of a **group**, given a certain number of their mutual **differences** (or **ratios**). (with input some ratios)



Graph representation:

Nodes = unknowns

Edges = observations

Consistency Constraint

$$Z_{ij} = X_i * X_j^{-1}$$

F. Arrigoni, A. Fusiello. *Synchronization problems in Computer Vision with closed-form solutions*. International Journal of Computer Vision (2020)

Problem Definition

we can generalize considerations done before for example...

🤔 Is the solution **unique**?

The solution is defined up to a global **group element**:

as adding same value to all the node difference does NOT change... here the same with multiplication

$$Z_{ij} = X_i * X_j^{-1} = X_i * (S * S^{-1}) * X_j^{-1} = (X_i * S) * (X_j * S)^{-1}$$

CONSTRAINT STILL SATISFYED

multiply by same value!

Consistency constraint

Identity (compensated)

rewrite

New group elements (change of variables)

we are changing global reference system by multiply all nodes

👉 The **ambiguity** can be fixed by arbitrarily choosing one node (e.g. node 1) and assigning it to the **identity**.

↑ solution NOT unique... global element ~ AMBIGUITY!

NO ISSUE, we fix one node and assign I ↙ (IMPORTANT to fix one node!)

to it, fixing GLOBAL Reference System

⇒ How to compute solution

same procedure, by TREE computation

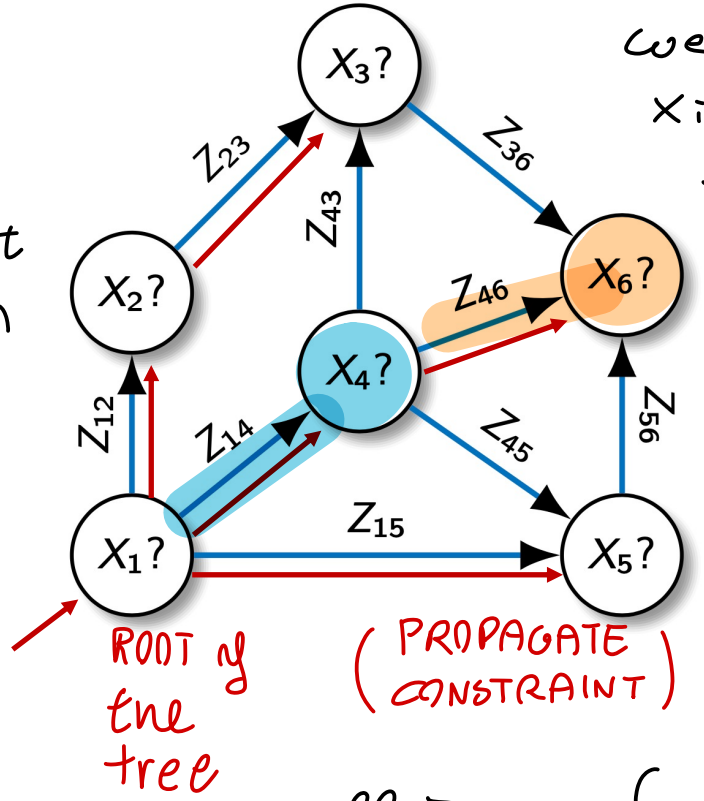
Problem Definition

A solution can be found via sequential propagation along a spanning tree.

arbitrary tree

More suitable to use the tree because give directly X_i (explict valued X_i assuming X_j is known)

We solve it by constraint propagation



ROOT of the tree

(PROPAGATE CONSTRAINT)

we get all X_i following the tree

all Z_{ij} are known ...

Equivalent Expression for the Consistency Constraint

$$Z_{ij} = X_i * X_j^{-1} \iff Z_{ij} * X_j = X_i$$

(KNOWN by assumption)

(Equivalent Reformulation)

$X_1 = 1_\Sigma$ arbitrary first node value

$X_4 = Z_{41} * X_1 = Z_{41}$ (I)

$X_6 = Z_{64} * X_4 = Z_{64} * Z_{41}$

... ITERATE procedure!

👉 Each measure satisfies: $Z_{ji} = Z_{ij}^{-1}$

easy to get all directions

← issue... we are computy sequentially! PROPAGATING the error!

we see a different solution

Problem Definition

{ solution subject to error propagation }

❌ The spanning tree solution is subject to **error accumulation**.

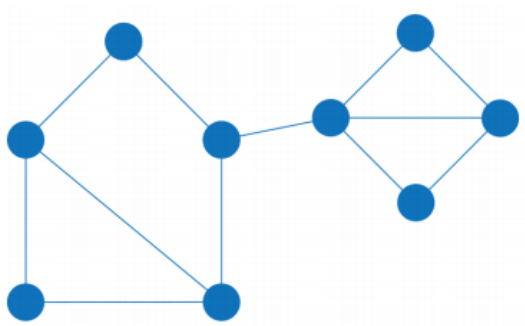
✅ The key component for achieving error compensation is to exploit **redundancy**, namely considering the whole graph.

when usimy the TREE approach we were NOT usimy All constraints, we skip some edges!

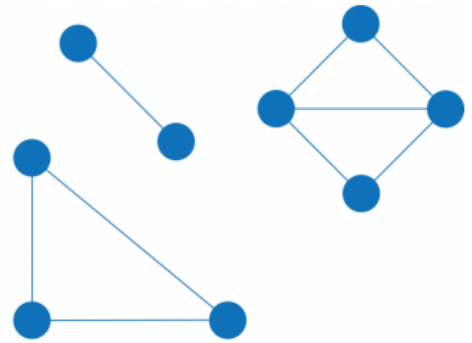
🤔 Under which assumption synchronization is **well-posed**?

The graph must be **connected** (= there is a path between any pair of nodes).

already assumed!



Connected Graph ✅



Disconnected Graph
Includes 3 components. ❌

so we find a SUBOPTIMAL solution, NOT all information are beeng used

(also a tree approach cannot be used if the GRAPH is disconnected! No way to reach all nodes... → we make that assumption for well posed problem

Outline

- Background
- Introduction
- **Spectral solution**
- Application to image mosaicking
- Conclusion & extensions
- Matlab demo

↗ better than sequential propagation !
EXPLOIT all the GRAPH ⟶

let's see the solution by considering INVERTIBLE MATRICES... we see what changes by working with homography

Problem Formulation

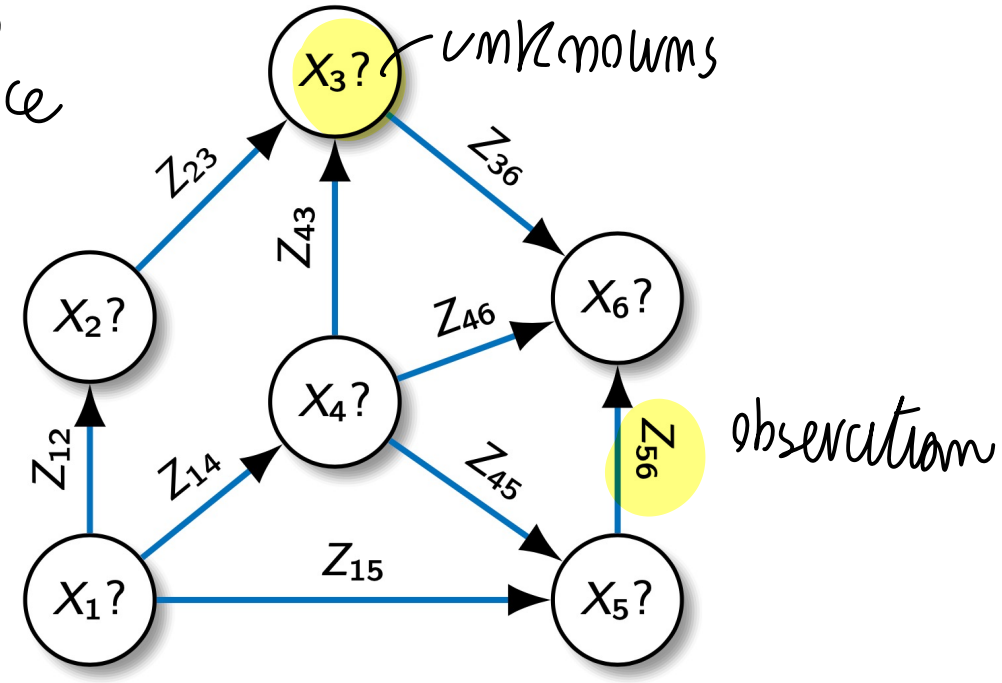
Let us consider the **General Linear Group** with matrix multiplication:

↑ this is the set of INVERTIBLE MATRIX

$$GL(d) = \{M \in \mathbb{R}^{d \times d} \text{ such that } \det(M) \neq 0\}$$ (invertible) ⟶ d x d invertible matrices

if any d dimension, we focus on 3x3 later

We work on this space



unknowns

observation

Graph representation:

- Nodes = unknowns
- Edges = observations

Consistency Constraint

$$Z_{ij} = X_i X_j^{-1}$$

connection node/edges follow constraint

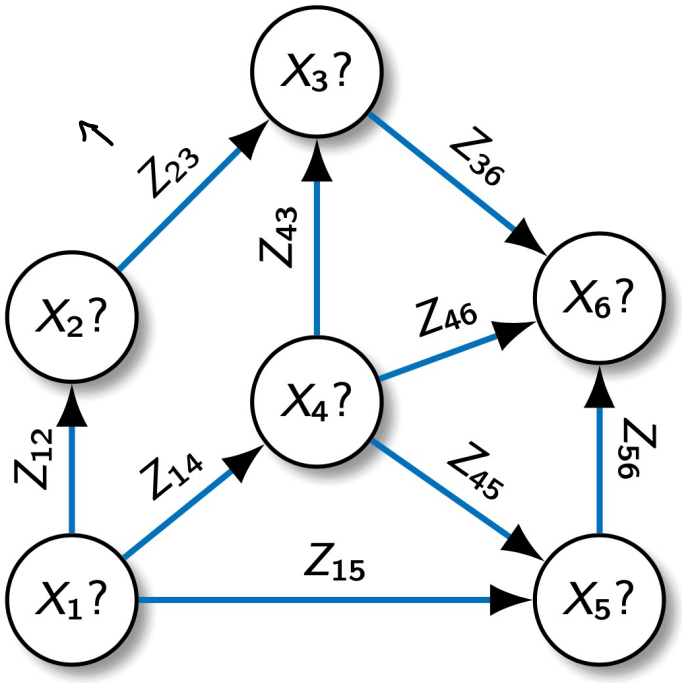
IMPLICATION...
in practice, many equations in the whole graph

Problem Formulation

The consistency constraint holds for a **single edge** in the graph: $Z_{ij} = X_i X_j^{-1}$

When considering the **entire graph**, we have multiple equations:

each edge gives rise to an equation of same form...
↓
(same structure)



for any edge..

known unknown

$$Z_{12} = X_1 X_2^{-1}$$
$$Z_{23} = X_2 X_3^{-1}$$
$$Z_{14} = X_1 X_4^{-1}$$
$$\cdots$$

for GLOBAL resolution, we collect all in a big matrix. Simplify computation

The case of a complete graph

The consistency constraint holds for a **single edge** in the graph: $Z_{ij} = X_i X_j^{-1}$

When considering the **entire graph**, an equivalent compact form can be obtained.

Let us consider the auxiliary variable: $Y = [X_1^{-1} \quad X_2^{-1} \quad \dots \quad X_n^{-1}]$ → Unknown, d x dn

collect all edges in a matrix similar to adjacency matrix... (bigger one)

JUST INVERSE

input in a single matrix

$$Z = \begin{bmatrix} I_d & Z_{12} & \dots & Z_{1n} \\ Z_{21} & I_d & \dots & Z_{2n} \\ \dots & & & \dots \\ Z_{n1} & Z_{n2} & \dots & I_d \end{bmatrix}$$

Known, dn x dn

each term is a block

d := size of block
n := # of nodes

$$X = \begin{bmatrix} X_1 \\ X_2 \\ \dots \\ X_n \end{bmatrix}$$

Unknown, dn x d

collect all unknowns in a MATRIX

$Z = XY$

Z is low-rank: rank(Z)=d

example, in image d = 3

Z: 3n x 3n MATRIX

3n x 3n in mosaicing

The case of a complete graph

The consistency constraint holds for a **single edge** in the graph: $Z_{ij} = X_i X_j^{-1}$

When considering the **entire graph**, an equivalent compact form can be obtained.

Let us consider the auxiliary variable: $Y = \begin{bmatrix} X_1^{-1} & X_2^{-1} & \dots & X_n^{-1} \end{bmatrix}$ → Unknown d x dn

element block (1,2) correspond to edge between 1ST and 2ND node

identity in the Diagonal ↓ (identity itself transform)

$$Z = \begin{bmatrix} I_d & Z_{12} & \dots & Z_{1n} \\ Z_{21} & I_d & \dots & Z_{2n} \\ \dots & & & \dots \\ Z_{n1} & Z_{n2} & \dots & I_d \end{bmatrix} \qquad X = \begin{bmatrix} X_1 \\ X_2 \\ \dots \\ X_n \end{bmatrix}$$

Known dn x dn

edge between 1ST and mTH node (last)

Unknown dn x d

⇑ Here is a COMPLETE graph (all nodes edged)

simplify equation constraint of all the graph...

$Z = XY$ — compact notation!

Z is low-rank: rank(Z)=d

we can deduce from Z that it is low rank... ~3 for homography very low respect dimension

The case of a complete graph

Known ← $Z = \begin{bmatrix} I_d & Z_{12} & \dots & Z_{1n} \\ Z_{21} & I_d & \dots & Z_{2n} \\ \dots & & & \dots \\ Z_{n1} & Z_{n2} & \dots & I_d \end{bmatrix}$ $X = \begin{bmatrix} X_1 \\ X_2 \\ \dots \\ X_n \end{bmatrix}$ → Unknown

⇓ PRACTICAL way to recover X from Z (based on SPECTRAL PROPERTY)

Proposition. *The columns of X are d eigenvectors of Z with eigenvalue n, where n is the number of nodes.*

Proof.

$$ZX = \underbrace{XY}_{\text{Consistency constraint}} X = X \underbrace{\sum_{i=1}^{n} (X_i^{-1} X_i)}_{X_i^{-1} X_i = I} = nX$$

($Z = XY$) ; $= YX$ written in this way

simply PROVEN by computations !

$$\boxed{ZX = nX}$$

from definition of Eigenvector

- A. Singer. *Angular synchronization by eigenvectors and semidefinite programming*. Applied and Computational Harmonic analysis (2011).
- M. Arie-Nachimson, S. Kovalsky, I. Kemelmacher-Shlizerman, A. Singer, R. Basri. *Global motion estimation from point matches*. 3DIM/3DPVT (2012).

The case of missing data

↓ it is possible to approach the problem when we have a GENERAL graph, NOT complete!

↓ to modify Z when data missing, you put 0 where data are missing

Known ← $Z = \begin{bmatrix} I_d & Z_{12} & \dots & 0 \\ Z_{21} & I_d & \dots & Z_{2n} \\ \dots & & & \dots \\ 0 & Z_{n2} & \dots & I_d \end{bmatrix}$ $\quad X = \begin{bmatrix} X_1 \\ X_2 \\ \dots \\ X_n \end{bmatrix}$ → Unknown

Measure of the pair (1,n) is missing

same size as before..

RESULTS connecting Results ↓

👉 Missing measures translate into **zero blocks** in Z.

{previously $ZX = mX$}

↓

Proposition. *The unknown matrix X satisfies the following equalities.*

different equality!

this can be rewritten as eigenvalue properties

$$ZX = (D \otimes I_d)X \iff \underbrace{(D \otimes I_d)^{-1} Z}_{W} X = X$$

($\otimes$: KRONEKER PRODUCT)

Degree matrix n x n (as seen before)

Scaled measurement matrix

The case of missing data

Known ⟵ $Z = \begin{bmatrix} I_d & Z_{12} & \dots & 0 \\ Z_{21} & I_d & \dots & Z_{2n} \\ \dots & & & \dots \\ 0 & Z_{n2} & \dots & I_d \end{bmatrix}$ $\quad X = \begin{bmatrix} X_1 \\ X_2 \\ \dots \\ X_n \end{bmatrix}$ ⟶ Unknown

Measure of the pair (1,n) is missing

👉 Missing measures translate into **zero blocks** in Z.

Proposition. *The unknown matrix X satisfies the following equalities.*

$$(*)\ ZX = (D \otimes I_d)X \iff \underbrace{(D \otimes I_d)^{-1} Z}_{W} X = X$$

(inverting)

Degree matrix n x n

Scaled measurement matrix

scaled version of our input measurement matrix!

similar to previous case $ZX = WX$

The case of missing data

Known ⟵ $Z = \begin{bmatrix} I_d & Z_{12} & \dots & 0 \\ Z_{21} & I_d & \dots & Z_{2n} \\ \dots & & & \dots \\ 0 & Z_{n2} & \dots & I_d \end{bmatrix}$ $\quad X = \begin{bmatrix} X_1 \\ X_2 \\ \dots \\ X_n \end{bmatrix}$ ⟶ Unknown

Measure of the pair (1,n) is missing

👉 Missing measures translate into **zero blocks** in Z.

Proposition. *The columns of X are d eigenvectors of the matrix* $W = (D \otimes I_d)^{-1} Z$ *with eigenvalue 1, which is the largest eigenvalue.*

$WX = 1 \times$ (largest eigenvalue } NOT biggerone!

$$WX = X$$

with # of eigenvectors as the size of X (and each block)

$Zx = nx$, $\frac{1}{n} Zx = x$, $Wx = x$ } can generalize this case

EXAMPLE of KRONEKER ↓

The case of missing data

🤔 Who is the matrix $D \otimes I_d$? ↓ example of Kroneker of this MATRIX...

$$D = \begin{matrix} 2 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 3 \end{matrix} \qquad I_2 = \begin{matrix} 1 & 0 \\ 0 & 1 \end{matrix} \Rightarrow \quad D \otimes I_2 = \begin{matrix} 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 \end{matrix}$$

(simple computation)

↑
used to obtain
big matrix of
dim as input, with deg copy

Algorithm

The **spectral method** for **synchronization** can be detailed as follows:

$$W = (D \otimes I_d)^{-1} Z$$

$$Z = \begin{bmatrix} I_d & Z_{12} & \dots & 0 \\ Z_{21} & I_d & \dots & Z_{2n} \\ \dots & & & \dots \\ 0 & Z_{n2} & \dots & I_d \end{bmatrix}$$

Compute Eigenvectors

$$U = \begin{bmatrix} U_1 \\ U_2 \\ \dots \\ U_n \end{bmatrix}$$

Fix group ambiguity

$$X = U U_1^{-1} = \begin{bmatrix} I_d \\ U_2 U_1^{-1} \\ \dots \\ U_n U_1^{-1} \end{bmatrix}$$

(NOT Full eigenvalue decomposition)

- We do not need to compute all the eigenvectors, but only the d **leading** ones.
- The scaled measurement matrix has size dn x dn: **sparse** eigen-solvers can be exploited (Matlab `eigs`) to manage large-scale scenarios with missing data.

F. Arrigoni, B. Rossi, A. Fusiello. *Spectral synchronization of multiple views in SE(3)*. SIAM Journal on Imaging Sciences (2016).

for a general graph / complete graph

rank(Z) = d When d = 3 → rank (Z) = 3

It has only d = 3 non zeros singular values ←

which are the bigger ones!

↑

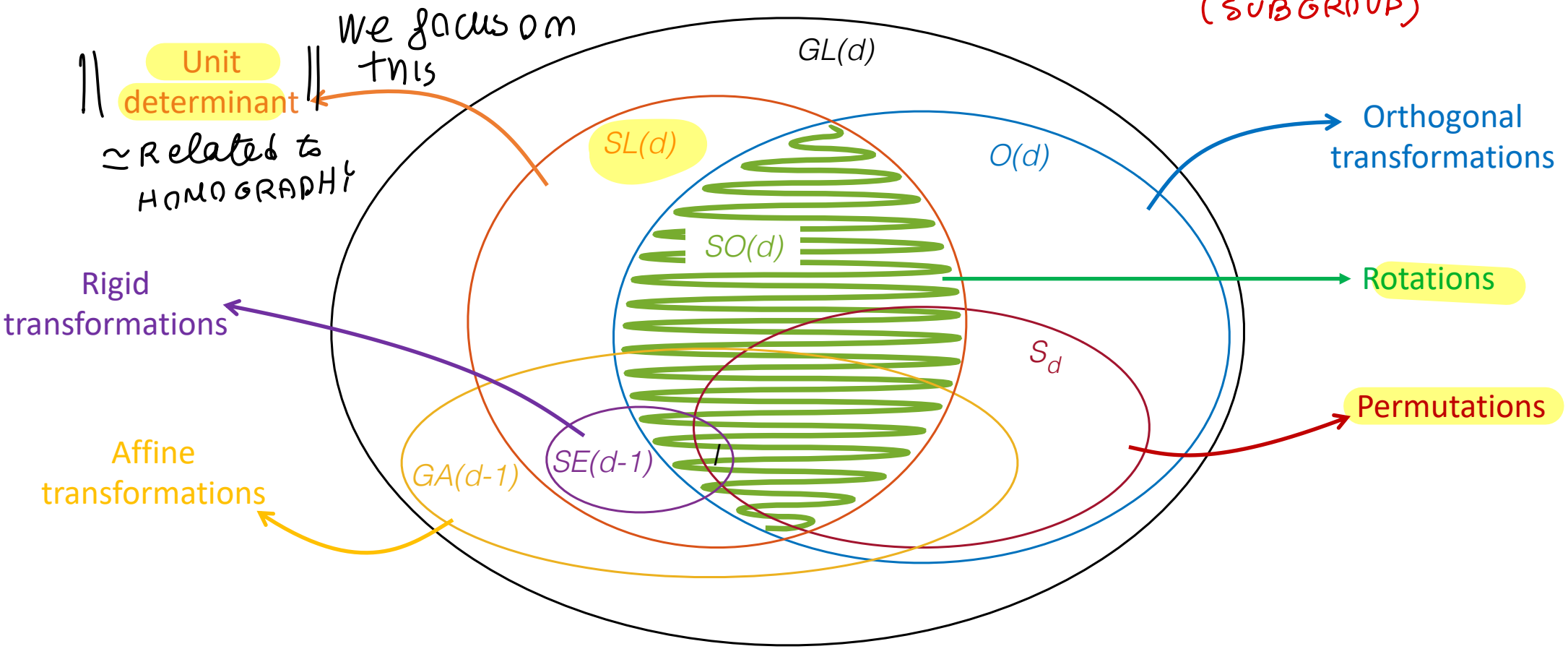
this is proven in a complete graph

← we need few eigenvectors, NOT all, just d eigs! (use sparse solver)

Algorithm

⇓ can be applyed to many subset of invertible matrix set (SUBGROUP)

The spectral method works for any **subgroup** of GL(d).

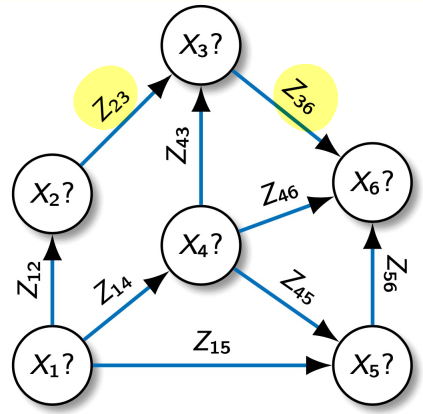


Algorithm

with respect to general case...

The spectral method works for any **subgroup** of GL(d).

Input:
graph with observations



1

Store observations in a **block-matrix** where zero blocks correspond to missing measures.

$$Z = \begin{bmatrix} I_d & Z_{12} & \dots & 0 \\ Z_{21} & I_d & \dots & Z_{2n} \\ \dots & & & \dots \\ 0 & Z_{n2} & \dots & I_d \end{bmatrix}$$

2

Rescale the measurement matrix using the degree matrix of the graph.

$$W = (D \otimes I_d)^{-1} Z$$

3

Compute the d leading **eigenvectors.**

$$U = \begin{bmatrix} U_1 \\ U_2 \\ \dots \\ U_n \end{bmatrix}$$

4

Fix the group **ambiguity**: the first element is set to the identity.

$$V = \begin{bmatrix} I_d \\ V_2 \\ \dots \\ V_n \end{bmatrix}$$

5

Project the solution onto the group.

$$X = \begin{bmatrix} I_d \\ X_2 \\ \dots \\ X_n \end{bmatrix}$$

in subset, a general PROJECTION is necessary to avoid noise...

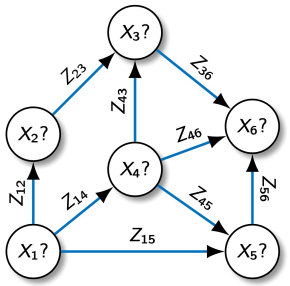
Algorithm

additional step projecty solution

⇓

GENERAL FLOW

Graph with observations on the edges



Compute eigenvectors OF MATRIX

Use existing solvers (Matlab eigs)

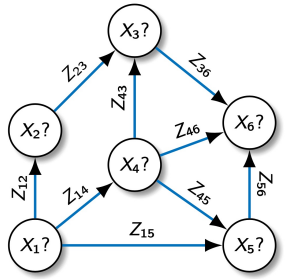
Fix ambiguity

Right-multiply by the inverse of the first block

Final projection

It depends on the group

Values on the nodes



👉 At the end the solution needs to be projected onto the group: the spectral method does not enforce specific **constraints** of group elements.

Outline

- Background
- Introduction
- Spectral solution
- **Application to image mosaicking**
- Conclusion & extensions
- Matlab demo

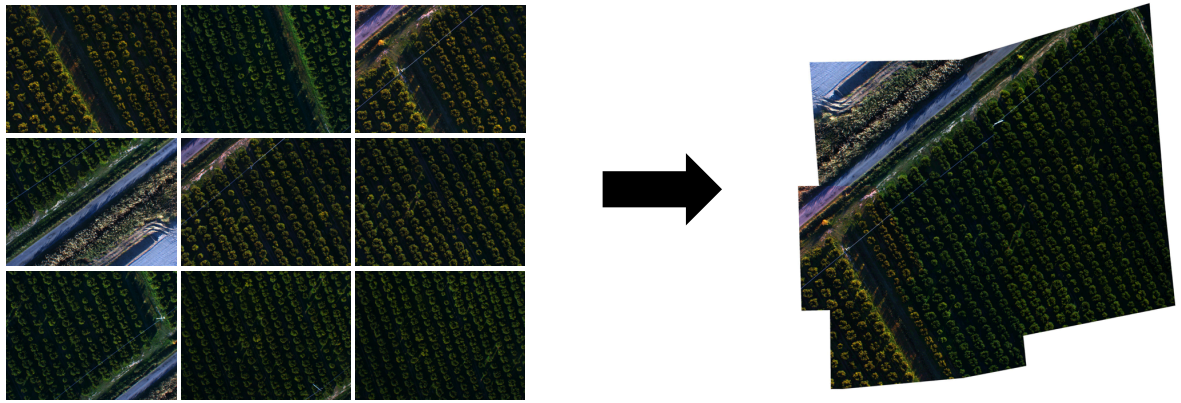
application...

Problem Formulation

TASK:

Image mosaicking: the task is to align multiple images into a mosaic.

(Many images!)



The underlying transformations are **homographies:**

- The scene is (approximately) planar
- The camera is rotating only (PURE ROTATION)

P. Schroeder, A. Bartoli, P. Georgel, N. Navab. *Closed- form solutions to multiple-view homography estimation*. WACV (2011).

a TRICK allow to make it $Z_{ij} = X_i X_j^{-1}$ by choosing proper NORMALIZATION

Problem Formulation

How can we deal with the **scale ambiguity**?

$$Z_{ij} \simeq X_i X_j^{-1}$$

Equality up to scale

more difficult problem!

Additional PROBLEM...
the MATRIX is defined up to scale!

in CONSISTENCY constraint we don't have "=" but "$\simeq$ up to scale"

We need to properly choose a normalization:

- We would like to have a **strict equality**, in order to apply the synchronization framework (spectral solution).

(we seen before!)

- We need a normalization that preserves **group** properties (e.g. multiplication of two group elements belongs to the group).

we have SCALE AMBIGUITY issue → we remove this by proper normalization!

Problem Formulation

Homographies can be conveniently represented as matrices with **unit determinant,** which form the **Special Linear Group** SL(3):

this is usefull, (normalize to det = 1. is usefull) this ensure the set is a group, results has still det = 1

$$SL(3) = \{M \in \mathbb{R}^{3\times 3} \text{ such that } \det(M) = 1\}$$

Fix scale ambiguity

Just normalizing remove up to scale and ensure work with constraint

- Closure: $\forall A, B \in SL(3): \quad \det(AB) = \det(A)\det(B) = 1 \implies AB \in SL(3)$
- Identity: $\det(I_3) = 1 \implies I_3 \in SL(3)$
- Inverse: $\forall A \in SL(3): \quad \det(A^{-1}) = \det(A)^{-1} = 1 \implies A^{-1} \in SL(3)$

→ Properties of determinant

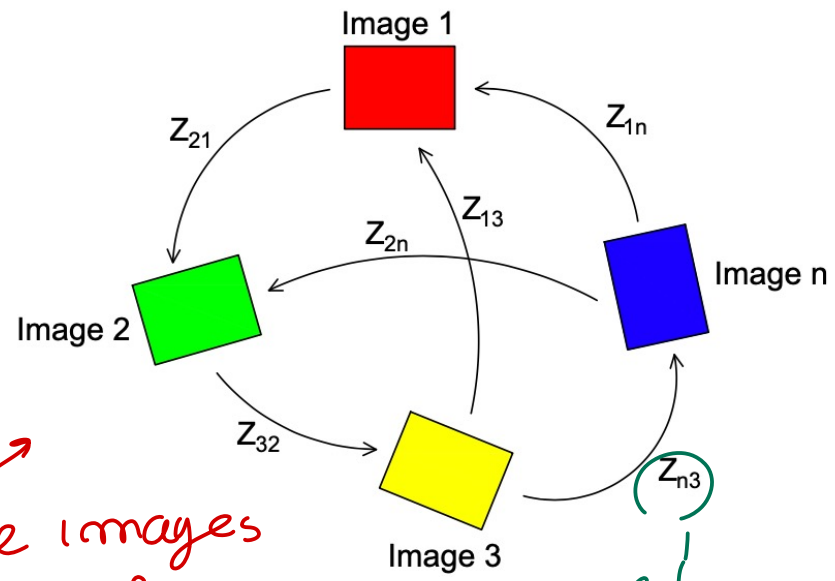
mosaicking
↓
~ solving in 2 steps!
• compute transformation brings one image into another (with 2 images (homography))

Two-step Approach

The problem can be solved in **two steps**:

1. Align **pairs** of images in isolation;
2. Compute a **global** alignment starting from pairwise transformations.

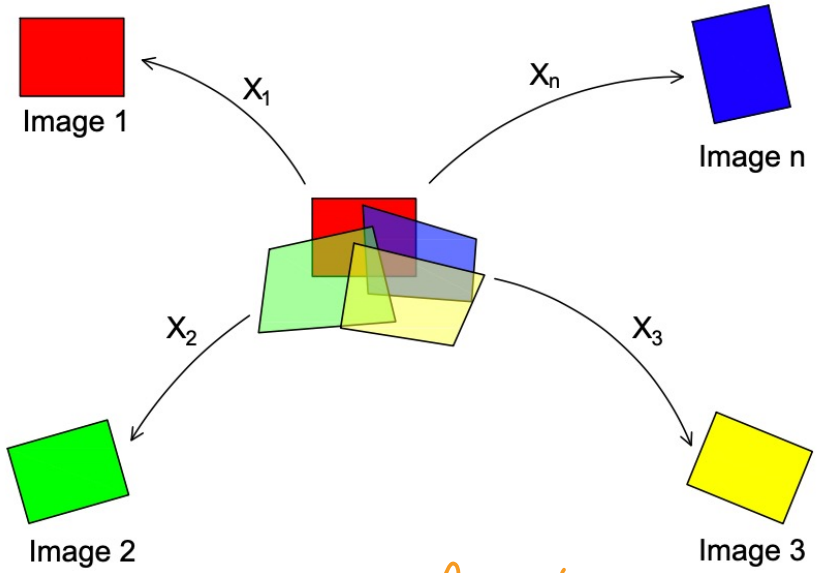
When many images compute all independently of all pairs



encode images in a graph

each node image

edges are pairwise homography



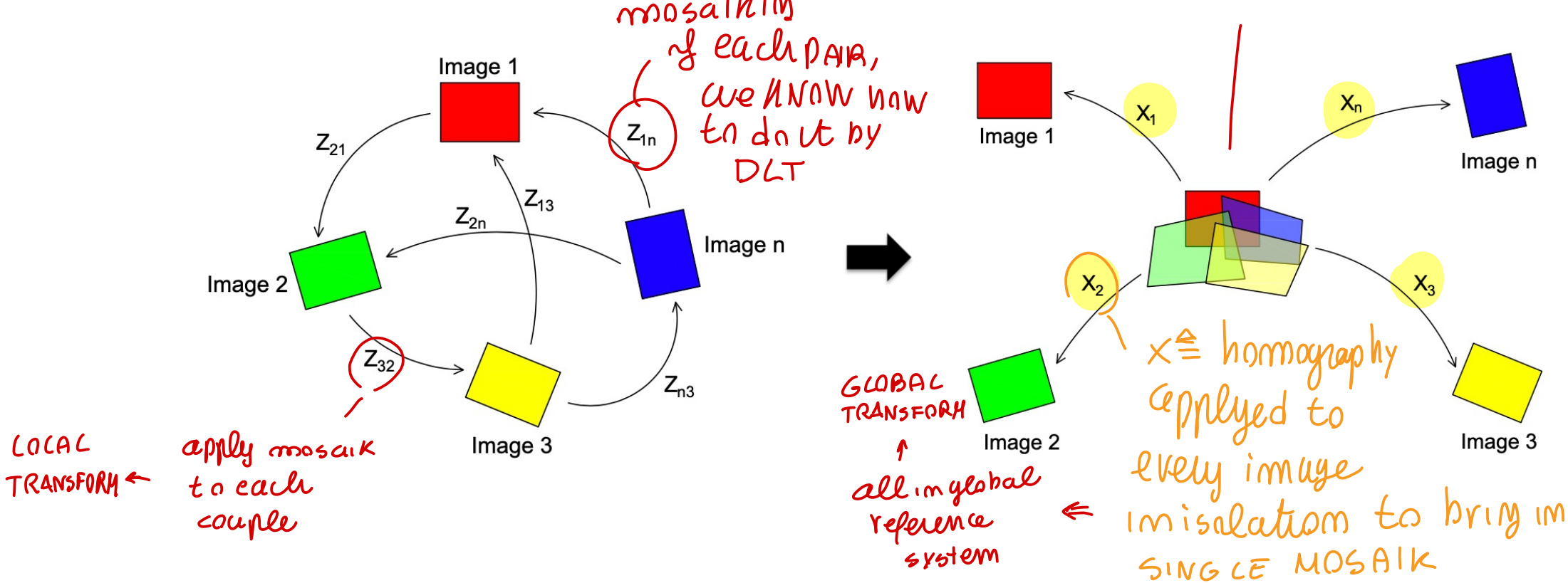
you already know HOW TO compute couples!

Two-step Approach

Global task: compute transform for each image that brings all in a unique mosaik

The problem can be solved in **two steps**:

1. Align **pairs** of images in isolation;
2. Compute a **global** alignment starting from pairwise transformations.



When considering just 2 images

↓

DLT to compute homography by correspondancy

Direct Linear Transform

IF you have pairs of images... ⇒

∀ pairs of images → solve mosaicking by DLT
BUT how to global align by PAIRWISE noisy transformation ? ⇒ remove noise, have global error compensation while putting in global reference system

Two-step Approach

The problem can be solved in **two steps**:

1. Align **pairs** of images in isolation; (LOCALS)
2. Compute a **global** alignment starting from pairwise transformations.

↳ absolute !

👉 **Remarks:**

- In Step 1 the transformations are **local/relative** whereas in Step 2 they are **global/absolute.**
- Correspondences are used in Step 1 only, but they are not used in Step 2.

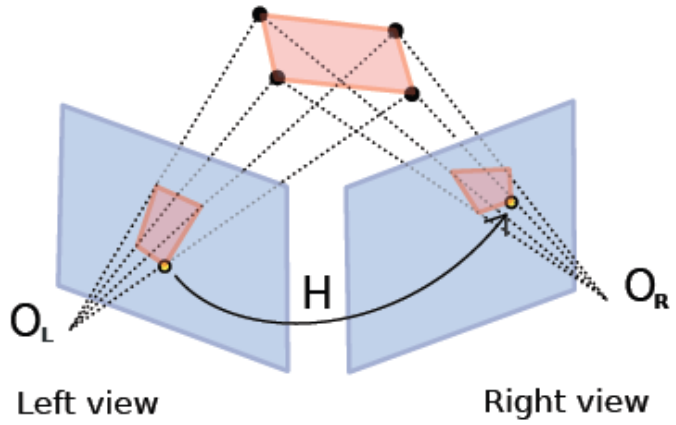
↓ points correspondance only 1ST

↑ only works in terms of transformation, NOT WITH POINTS

Two-step Approach

The problem can be solved in **two steps**:

1. Align **pairs** of images in isolation;
2. Compute a **global** alignment starting from pairwise transformations.



STEP 1

Standard techniques can be used:

- Compute the **homography** from image point correspondences (DLT).
- Scale the homography to unit determinant, to get an element in SL(3).

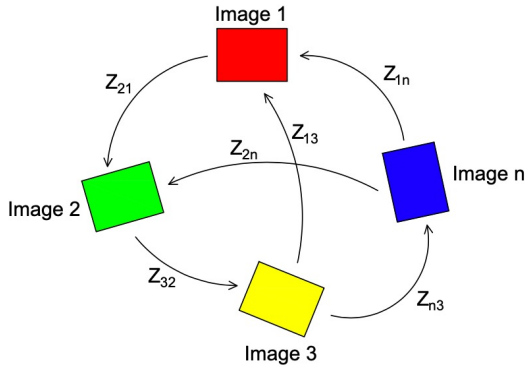
to normalize matrix

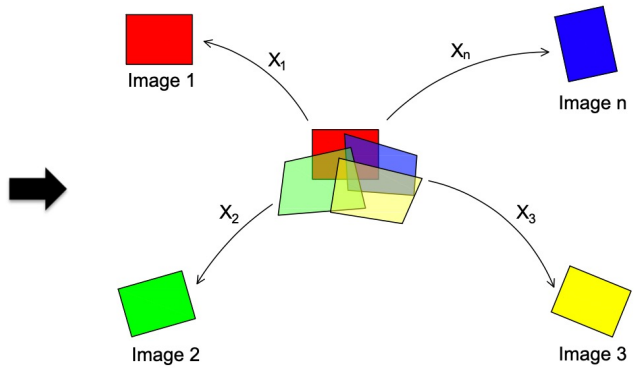
R. Hartley, A. Zisserman. *Multiple view geometry in computer vision*. Cambridge University Press, second edition (2004)

Two-step Approach

The problem can be solved in **two steps**:

1. Align **pairs** of images in isolation;
2. Compute a **global** alignment starting from pairwise transformations.





STEP 2

This is a synchronization problem in the Special Linear Group SL(3):

$$Z_{ij} = X_i X_j^{-1}$$

with framework described!

P. Schroeder, A. Bartoli, P. Georgel, N. Navab. *Closed- form solutions to multiple-view homography estimation*. WACV (2011).

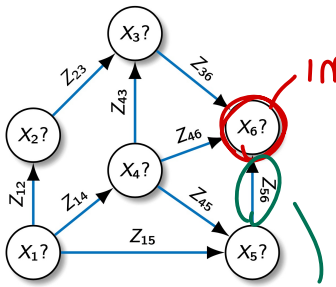
how it works for images Mosaicing

Spectral Solution

we can use previous framework!

comply with convention by scaling

Graph with observations on the edges → Compute eigenvectors → Fix ambiguity → Final projection → Values on the nodes



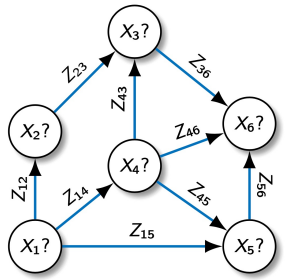
image

pairwise homography

Use existing solvers (Matlab eigs)

Right-multiply by the inverse of the first block

It depends on the group



rescale to have det = 1

👉 Projection can be done by dividing H by $\sqrt[3]{\det(H)}$, which has unit determinant.

In general: $\det(aU) = a^3 \det(U)$

we have scale ^3

det does not preserve scale for 3×3 matrix

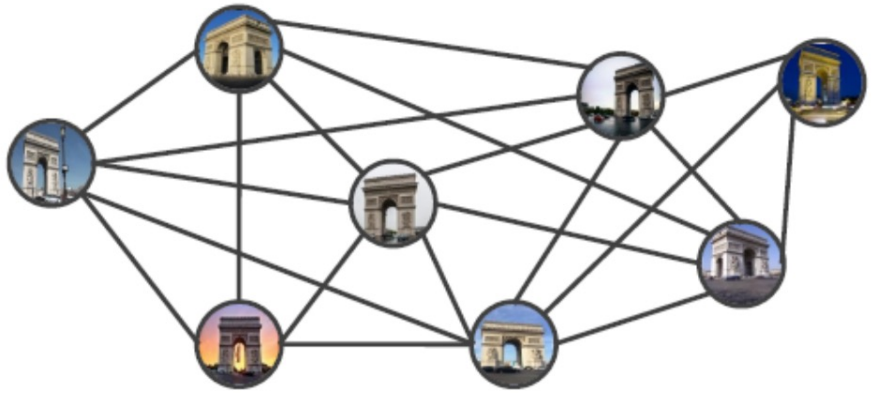
Outline

- Background
- Introduction
- Spectral solution
- Application to image mosaicking
- **Conclusion & extensions** → of this framework! ⇒
- Matlab demo

Extensions

There are many applications of synchronization in Computer Vision!

👉 The key structure to represent these problems is a **graph**.



common pattern and deffrent stuff

👉 All the variables are elements of a **group** (rotations, permutations, ...).

(matrix often)

Extensions

There are many applications of synchronization in Computer Vision!

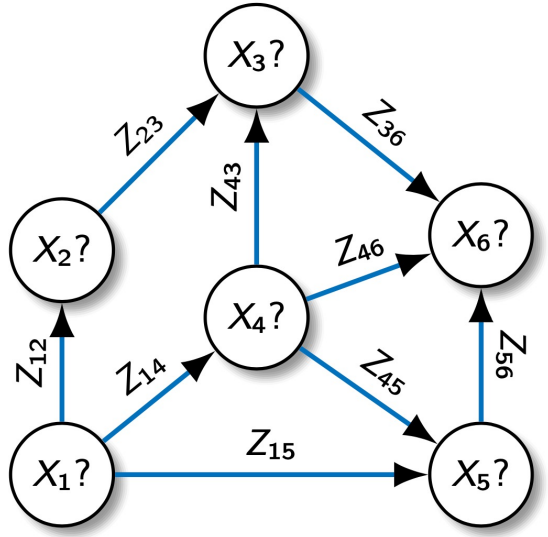
🤔 *What changes between them?*

- Tool used to get measures on edges
- Group (matrices) where the problem is formulated → projection

↳ impact final projection

🤔 *What remains the same?*

- Tool used to recover unknowns on the nodes → spectral method



Extensions

There are many applications of synchronization in Computer Vision!

🤔 *What changes between them?*

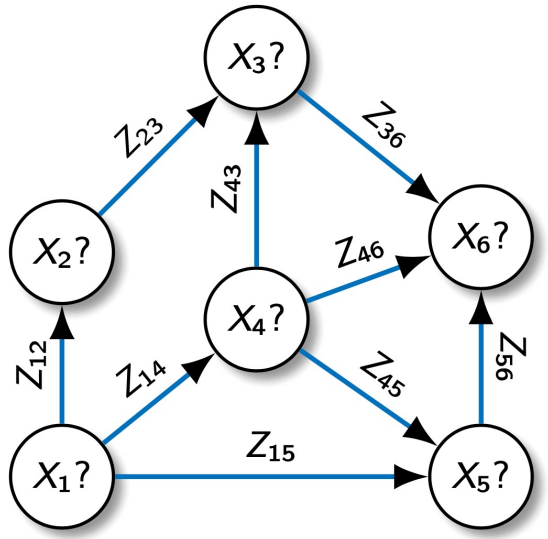
- Tool used to get measures on edges ⇢ Homography estimation
- Group (matrices) where the problem is formulated → projection
  - Matrices with unit determinant (Special Linear Group)
  - Scale to unit determinant

DLT to get homography

🤔 *What remains the same?*

- Tool used to recover unknowns on the nodes → spectral method

the same

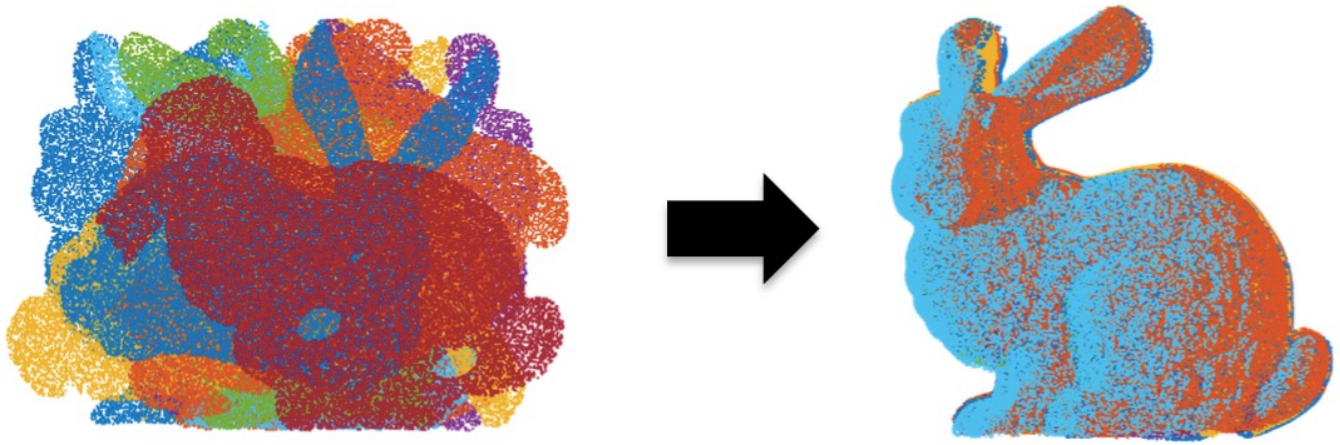


Other Scenarios

3D Registration

align point clouds into single shape

The task is to find the **rigid transformations** (rotations+translations) that bring multiple 3D point-clouds into alignment.



F. Arrigoni, B. Rossi, A. Fusiello. *Spectral synchronization of multiple views in SE(3)*. SIAM Journal on Imaging Sciences (2016)

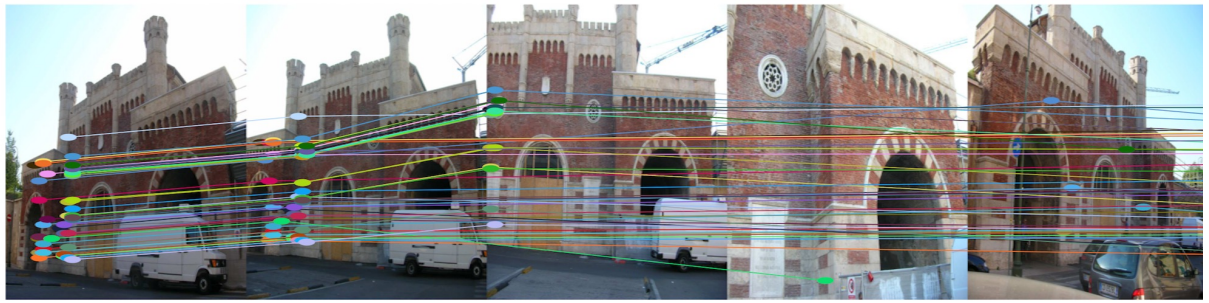
Other Scenarios

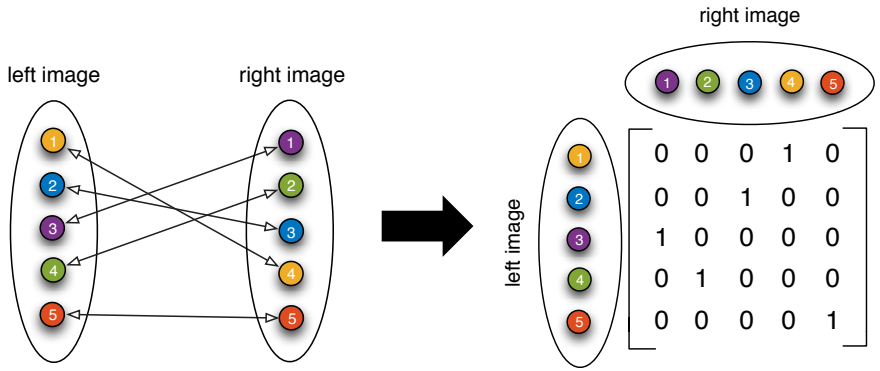
Multi-image Matching

by permutation matrix

correspondences between MANY images

The task is to find point correspondences between multiple images.





Matches can be represented as **permutation matrices**.

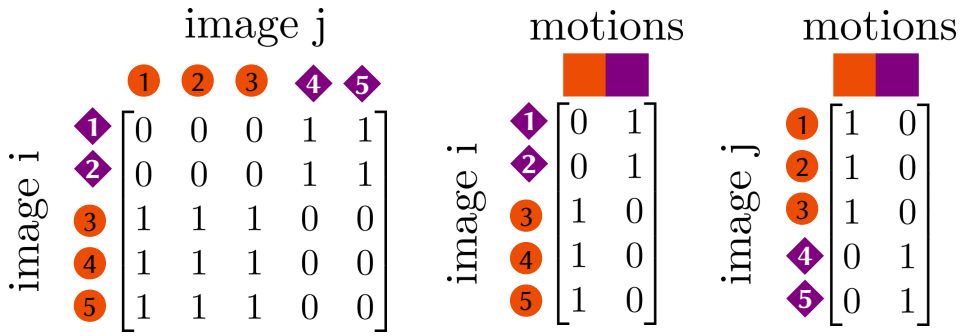
D. Pachauri, R. Kondor, V. Singh. *Solving the multi-way matching problem by permutation synchronization.* NIPS (2013)

Other Scenarios

Motion Segmentation

The goal is to classify points in images based on the **moving** object they belong to.

detect moving objects

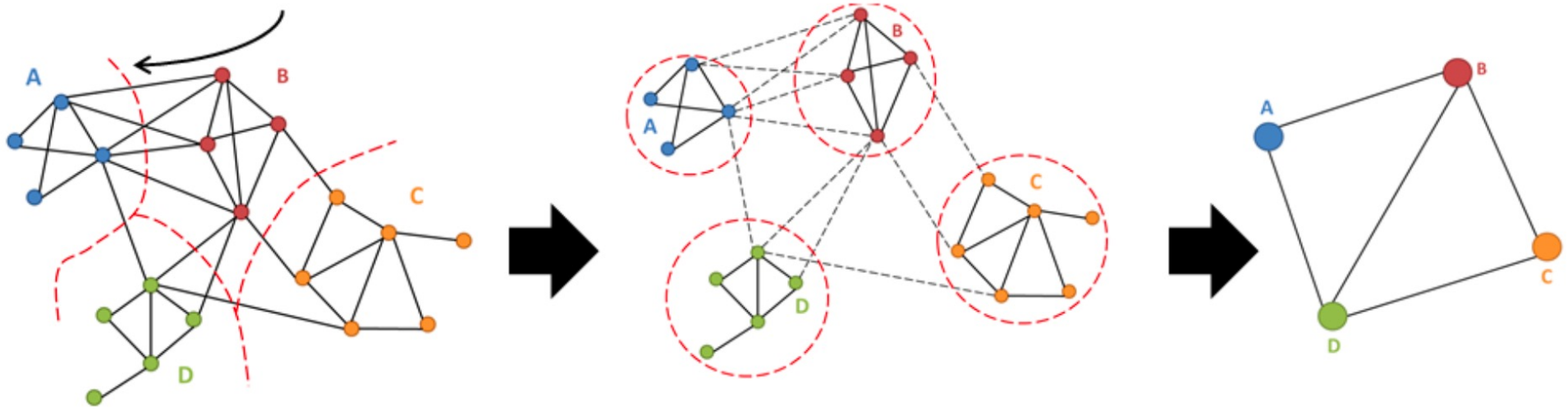


It can be viewed as a synchronization of **binary matrices**.

F. Arrigoni, T. Pajdla. *Motion segmentation via synchronization.* ICCV Workshops (2019)

Other Scenarios
Partitioned Synchronization

A synchronization task is partitioned into subproblems for the sake of efficiency.



Each subgraph has its own local **ambiguity**: finding consistency between the local ambiguities is an example of synchronization.

Bhowmick, Patra, Chatterjee, Govindu, Banerjee. *Divide and conquer: Efficient large-scale structure from motion using graph partitioning.* ACCV (2014)

Concluding remarks
Spectral Solution

❌ The spectral method is an **approximate** solution that does not enforce geometric constraints. (problem constraints not accounted)

✅ It entails a **simple** implementation: spectral decomposition + projection.

✅ It is **general**: it can be applied to any group admitting a matrix representation (rotations/rigid-motions/permutations/homographies) and also to sets that do not have the structure of a group (partial permutations/binary matrices).

👉 *There exist also other solutions, but they are specific for the chosen group.*

Concluding remarks

Synchronization

✅ The **two-step framework** permits to:

- exploit two-view tools, hence the problem is split into subproblems which are **easier** to solve (in closed-form sometimes);
- exploit a more **compact** representation as the problem is solved in frame-space (points are not considered after pairwise transformations);

❌ Results can be suboptimal as points are not considered.

✅ It promotes error compensation, being global.

✅ It involves a variety of **applications** in Computer Vision (registration, matching, mosaicking, motion segmentation, ...).

Outline

- Background
- Introduction
- Spectral solution
- Application to image mosaicking
- Conclusion & extensions
- **Matlab demo**

MATLAB Demo:

↓

1) you load image from path,
than import ALL images at one time
↓
the lenght is # images

2) resize for effcient code
↓
and save all images in cell

3) precompute matches.mat

⇓ SYNCHRONIZATION

(1) compute pairwise homography

1.1) Z = eye(3 * n_img) initialize, to I_{3n} so I on block diag

1.2) E error ausiliary variable! ⇝ you can use transfer_error_H
(as done in Image WARPING)

1.3) we find corresponding
matches

1.4) Preconditioning for numerical reason ~ transformation
to modify image coord

1.5) homography by DLT ⇒ we apply pre-conditioning
so than we inverse transform

↓

H_0 is V pair of images!
also H_0^{-1} is needed.

we don't compute
for repeated pairs

1.6) Put homography on right block... (of matrix Z)
as 3×3 blocks..

1.7) compute homography transfer error for additional analysis

We can compute SVD and visualize singular values to also see errors...

↓

eigenvalues in log scale
first are large
for $d > 3$ are $\sigma \sim 0$
We expect rank drop!
being noisy data, NOT ideal!

(2) synchronization

↓

we need Adjacency matrix!
then call function for synchronization

↓ solved by toolbox hom_sync.m function

using the procedure for synchronization described before!

numerical issue can occur, we extract IR values to avoid C results

⇓

we find all absolute homography

2) then apply H to all

by projtform2d(H{i}) $\forall i$

↑ (this mosaik all images)