Une approche approche par décomposition et reparamétrisation de la résolution de contraintes géométriques.

Rémi Imbach, Pascal Mathis et Pascal Schreck

Université de Strasbourg
LSIIT, UMR ULP-CNRS 7005

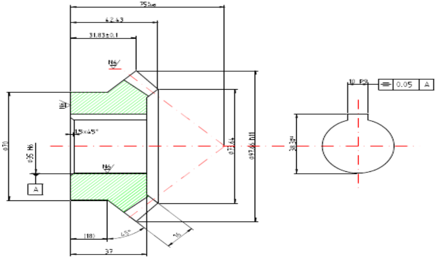
22 mars 2012

Résolution de contraintes et modélisation

- Modélisation géométrique
  - déclarative : décrire un objet par les propriétés qu'il satisfait ;
  - impérative : exhiber cet objet ou une façon de le construire.
- Résolution de Contraintes Géométriques (RCG) :

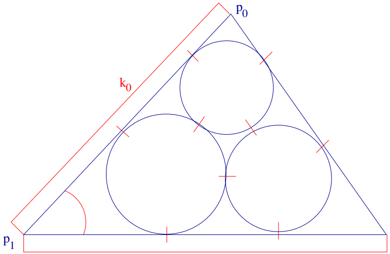
$$\text{Contraintes géométriques} \rightarrow \text{objet}$$

- Applications :
  - Chimie moléculaire
  - Robotique
  - CAO
  - ...



Systèmes de Contraintes Géométriques (SCG)

- Ensemble de primitives géométriques,
  - points, droites, cercles, ...
  - variables sortées : p_0, p_1 : *Point*
- satisfaisant un ensemble de contraintes :
  - angles, distances, tangences, ...
  - termes : $k_0 = dist_{pp}(p_0, p_1)$.



Inconnues :

$p_0, p_1, \ldots, c_0, c_1, \ldots$

Paramètres :

$k_0, k_1, \ldots$

Contraintes :

$k_0 = dist_{pp}(p_0, p_1)$

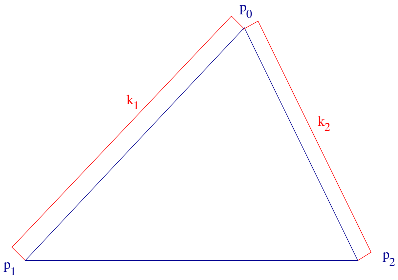
$k_1 = dist_{pp}(p_0, p_2)$

...

$tangent(c_0, c_1)$

...

Systèmes de Contraintes Géométriques (SCG)



- Un SCG est bien contraint :
  s'il admet un nombre fini de solutions.
- Un SCG est structurellement bien contraint :
  - SCG G avec n inconnues, c contraintes
    SCG $G' < G$ avec n' inconnues, c' contraintes
  - 2D : $c = 2n - 3$ et $c' \leq 2n' - 3$
  - 3D : $c = 3n - 6$ et $c' \leq 3n' - 6$

Résolution de SCG

Deux grandes familles de méthodes,

- celles utilisant un raisonnement géométrique :
  - Systèmes à Base de Connaissance (KBS),
  - Intersection des Lieux Géométriques (LIM),
- celles utilisant la géométrie analytique :

$$\text{SCG} \rightarrow \text{système d'équations}$$

  - résolu symboliquement (Ritt-Wu,bases de Gröbner,...),
  - ou numériquement (Bernstein, Newton-Raphson, homotopie).

et stratégie diviser pour règner.

Problématique :

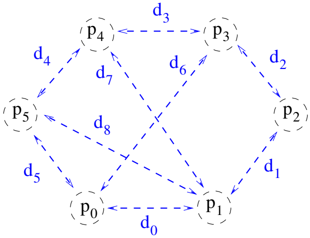
- palier les défauts des méthodes géométriques,
- guider une méthode numérique par la géométrie.

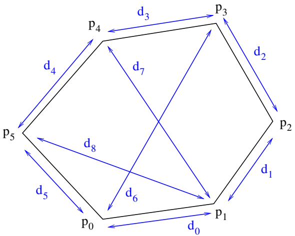
Méthode par Intersection des Lieux Géométriques (LIM)

- Obtention du plan de construction
  - Construire un hyper-graphe

sommets $\leftrightarrow$ primitives géométriques

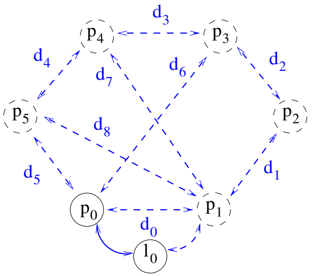
hyper-arêtes $\leftrightarrow$ contraintes





Méthode par Intersection des Lieux Géométriques (LIM)

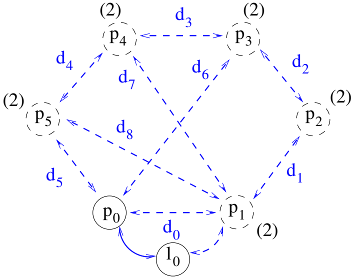
- Obtention du plan de construction
  - Construire un hyper-graphe
  - Fixer un *repère* pour localiser la construction :
    - ensemble de primitives déterminant un repère affine de l'espace
    - par exemple un point et une droite en 2D.



Parameters :
p_0, l_0
Unknowns :
.
Terms :
.

Méthode par Intersection des Lieux Géométriques (LIM)

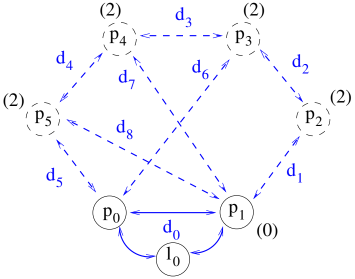
- Obtention du plan de construction
  - Construire un hyper-graphe
  - Fixer un *repère* pour localiser la construction :
  - Propagation avant des degrés de liberté



Parameters :
p_0, l_0
Unknowns :
.
Terms :
.

Méthode par Intersection des Lieux Géométriques (LIM)

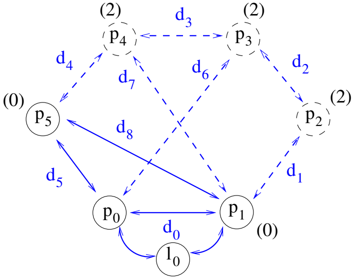
- Obtention du plan de construction
  - Construire un hyper-graphe
  - Fixer un *repère* pour localiser la construction :
  - Propagation avant des degrés de liberté



Parameters :
p_0, l_0, d_0
Unknowns :
c_0, p_1
Terms :
$c_0 = mkcircle(p_0, d_0)$
$p_1 = intercl(c_0, l_0)$

Méthode par Intersection des Lieux Géométriques (LIM)

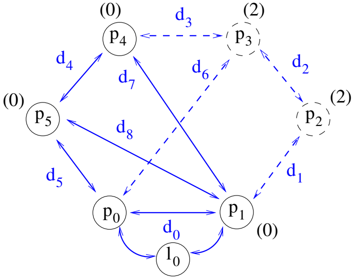
- Obtention du plan de construction
  - Construire un hyper-graphe
  - Fixer un *repère* pour localiser la construction :
  - Propagation avant des degrés de liberté



Parameters :
p_0, l_0, d_0, d_5, d_8
Unknowns :
c_0, p_1, c_5, c_8, p_5
Terms :
$c_0 = mkcircle(p_0, d_0)$
$p_1 = intercl(c_0, l_0)$
$c_5 = mkcircle(p_0, d_5)$
$c_8 = mkcircle(p_1, d_8)$
$p_5 = intercc(c_5, c_8)$

Méthode par Intersection des Lieux Géométriques (LIM)

- Obtention du plan de construction
  - Construire un hyper-graphe
  - Fixer un *repère* pour localiser la construction :
  - Propagation avant des degrés de liberté



Parameters :

$p_0, l_0, d_0, d_5, d_8, d_4, d_7$

Unknowns :

$c_0, p_1, c_5, c_8, p_5, d_4, d_7, p_4$

Terms :

$c_0 = mkcircle(p_0, d_0)$

$p_1 = intercl(c_0, l_0)$

$c_5 = mkcircle(p_0, d_5)$

$c_8 = mkcircle(p_1, d_8)$

$p_5 = intercc(c_5, c_8)$

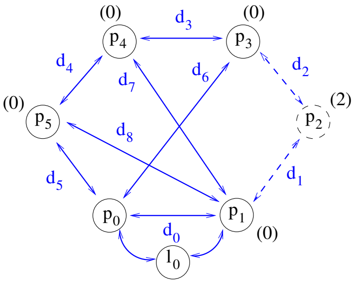
$c_4 = mkcircle(p_5, d_4)$

$c_7 = mkcircle(p_1, d_7)$

$p_4 = intercc(c_4, c_7)$

Méthode par Intersection des Lieux Géométriques (LIM)

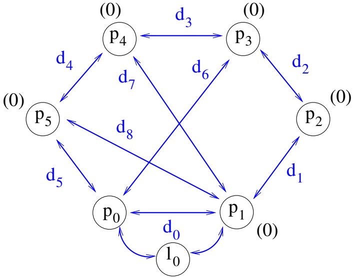
- Obtention du plan de construction
  - Construire un hyper-graphe
  - Fixer un *repère* pour localiser la construction :
  - Propagation avant des degrés de liberté



Parameters :
$p_0, l_0, d_0, d_5, d_8, d_4, d_7, d_3, d_6$
Unknowns :
$c_0, p_1, c_5, c_8, p_5, d_4, d_7, p_4, c_3, c_6, p_3$
Terms :
$c_0 = mkcircle(p_0, d_0)$
$p_1 = intercl(c_0, l_0)$
$c_5 = mkcircle(p_0, d_5)$
$c_8 = mkcircle(p_1, d_8)$
$p_5 = intercc(c_5, c_8)$
$c_4 = mkcircle(p_5, d_4)$
$c_7 = mkcircle(p_1, d_7)$
$p_4 = intercc(c_4, c_7)$
$c_3 = mkcircle(p_4, d_3)$
$c_6 = mkcircle(p_0, d_6)$
$p_3 = intercc(c_3, c_6)$

Méthode par Intersection des Lieux Géométriques (LIM)

- Obtention du plan de construction
  - Construire un hyper-graphe
  - Fixer un *repère* pour localiser la construction :
  - Propagation avant des degrés de liberté



Parameters :

$p_0, l_0, d_0, d_5, d_8, d_4, d_7, d_3, d_6, d_1, d_2$

Unknowns :

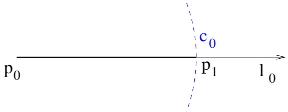
$c_0, p_1, c_5, c_8, p_5, d_4, d_7, p_4, c_3, c_6, p_3, c_1, c_2, p_2$

Terms :

$c_0 = mkcircle(p_0, d_0)$
$p_1 = intercl(c_0, l_0)$
$c_5 = mkcircle(p_0, d_5)$
$c_8 = mkcircle(p_1, d_8)$
$p_5 = intercc(c_5, c_8)$
$c_4 = mkcircle(p_5, d_4)$
$c_7 = mkcircle(p_1, d_7)$
$p_4 = intercc(c_4, c_7)$
$c_3 = mkcircle(p_4, d_3)$
$c_6 = mkcircle(p_0, d_6)$
$p_3 = intercc(c_3, c_6)$
$c_2 = mkcircle(p_3, d_2)$
$c_1 = mkcircle(p_1, d_1)$
$p_2 = intercc(c_1, c_2)$

Méthode par Intersection des Lieux Géométriques (LIM)

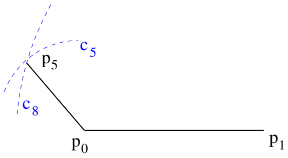
- Obtention du plan de construction
  - Construire un hyper-graphe
  - Fixer un *repère* pour localiser la construction :
  - Propagation avant des degrés de liberté
- Évaluation du plan de construction



Parameters :
$p_0, l_0, d_0, d_5, d_8, d_4, d_7, d_3, d_6, d_1, d_2$
Unknowns :
$c_0, p_1, c_5, c_8, p_5, d_4, d_7, p_4, c_3, c_6, p_3, c_1, c_2, p_2$
Terms :
$c_0 = mkcircle(p_0, d_0)$
$p_1 = intercl(c_0, l_0)$
$c_5 = mkcircle(p_0, d_5)$
$c_8 = mkcircle(p_1, d_8)$
$p_5 = intercc(c_5, c_8)$
$c_4 = mkcircle(p_5, d_4)$
$c_7 = mkcircle(p_1, d_7)$
$p_4 = intercc(c_4, c_7)$
$c_3 = mkcircle(p_4, d_3)$
$c_6 = mkcircle(p_0, d_6)$
$p_3 = intercc(c_3, c_6)$
$c_2 = mkcircle(p_3, d_2)$
$c_1 = mkcircle(p_1, d_1)$
$p_2 = intercc(c_1, c_2)$

Méthode par Intersection des Lieux Géométriques (LIM)

- Obtention du plan de construction
- Évaluation du plan de construction



Parameters :

$p_0, l_0, d_0, d_5, d_8, d_4, d_7, d_3, d_6, d_1, d_2$

Unknowns :

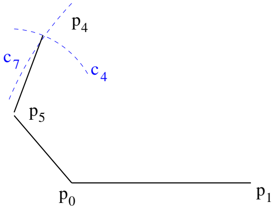
$c_0, p_1, c_5, c_8, p_5, d_4, d_7, p_4, c_3, c_6, p_3, c_1, c_2, p_2$

Terms :

$c_0 = mkcircle(p_0, d_0)$
$p_1 = intercl(c_0, l_0)$
$c_5 = mkcircle(p_0, d_5)$
$c_8 = mkcircle(p_1, d_8)$
$p_5 = intercc(c_5, c_8)$
$c_4 = mkcircle(p_5, d_4)$
$c_7 = mkcircle(p_1, d_7)$
$p_4 = intercc(c_4, c_7)$
$c_3 = mkcircle(p_4, d_3)$
$c_6 = mkcircle(p_0, d_6)$
$p_3 = intercc(c_3, c_6)$
$c_2 = mkcircle(p_3, d_2)$
$c_1 = mkcircle(p_1, d_1)$
$p_2 = intercc(c_1, c_2)$

Méthode par Intersection des Lieux Géométriques (LIM)

- Obtention du plan de construction
- Évaluation du plan de construction



Parameters :

$p_0, l_0\ , d_0\ , d_5, d_8\ , d_4, d_7\ , d_3, d_6\ , d_1, d_2$

Unknowns :

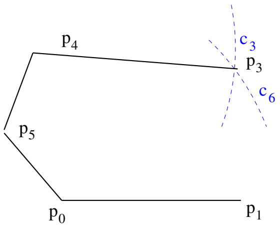
$c_0, p_1\ , c_5, c_8, p_5\ , d_4, d_7, p_4\ , c_3, c_6, p_3\ , c_1, c_2, p_2$

Terms :

$c_0 = mkcircle(p_0, d_0)$
$p_1 = intercl(c_0, l_0)$
$c_5 = mkcircle(p_0, d_5)$
$c_8 = mkcircle(p_1, d_8)$
$p_5 = intercc(c_5, c_8)$
$c_4 = mkcircle(p_5, d_4)$
$c_7 = mkcircle(p_1, d_7)$
$p_4 = intercc(c_4, c_7)$
$c_3 = mkcircle(p_4, d_3)$
$c_6 = mkcircle(p_0, d_6)$
$p_3 = intercc(c_3, c_6)$
$c_2 = mkcircle(p_3, d_2)$
$c_1 = mkcircle(p_1, d_1)$
$p_2 = intercc(c_1, c_2)$

Méthode par Intersection des Lieux Géométriques (LIM)

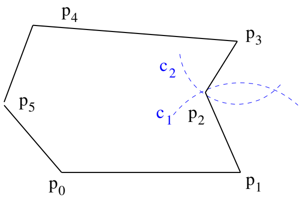
- Obtention du plan de construction
- Évaluation du plan de construction



Parameters :
$p_0, l_0, d_0, d_5, d_8, d_4, d_7, d_3, d_6, d_1, d_2$
Unknowns :
$c_0, p_1, c_5, c_8, p_5, d_4, d_7, p_4, c_3, c_6, p_3, c_1, c_2, p_2$
Terms :
$c_0 = mkcircle(p_0, d_0)$
$p_1 = intercl(c_0, l_0)$
$c_5 = mkcircle(p_0, d_5)$
$c_8 = mkcircle(p_1, d_8)$
$p_5 = intercc(c_5, c_8)$
$c_4 = mkcircle(p_5, d_4)$
$c_7 = mkcircle(p_1, d_7)$
$p_4 = intercc(c_4, c_7)$
$c_3 = mkcircle(p_4, d_3)$
$c_6 = mkcircle(p_0, d_6)$
$p_3 = intercc(c_3, c_6)$
$c_2 = mkcircle(p_3, d_2)$
$c_1 = mkcircle(p_1, d_1)$
$p_2 = intercc(c_1, c_2)$

Méthode par Intersection des Lieux Géométriques (LIM)

- Obtention du plan de construction
- Évaluation du plan de construction



Parameters :

$p_0, l_0, d_0, d_5, d_8, d_4, d_7, d_3, d_6, d_1, d_2$

Unknowns :

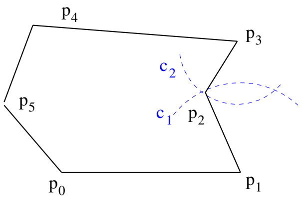
$c_0, p_1, c_5, c_8, p_5, d_4, d_7, p_4, c_3, c_6, p_3, c_1, c_2, p_2$

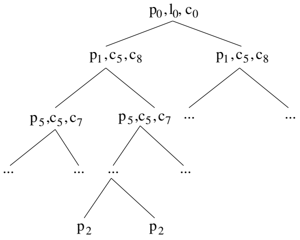
Terms :

$c_0 = mkcircle(p_0, d_0)$
$p_1 = intercl(c_0, l_0)$
$c_5 = mkcircle(p_0, d_5)$
$c_8 = mkcircle(p_1, d_8)$
$p_5 = intercc(c_5, c_8)$
$c_4 = mkcircle(p_5, d_4)$
$c_7 = mkcircle(p_1, d_7)$
$p_4 = intercc(c_4, c_7)$
$c_3 = mkcircle(p_4, d_3)$
$c_6 = mkcircle(p_0, d_6)$
$p_3 = intercc(c_3, c_6)$
$c_2 = mkcircle(p_3, d_2)$
$c_1 = mkcircle(p_1, d_1)$
$p_2 = intercc(c_1, c_2)$

Méthode par Intersection des Lieux Géométriques (LIM)

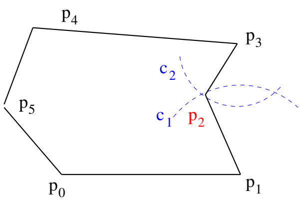
- Obtention du plan de construction
- Évaluation du plan de construction
- Arbre d'interprétation
  - Nombre exponentiel de branches en le nb d'intersections.

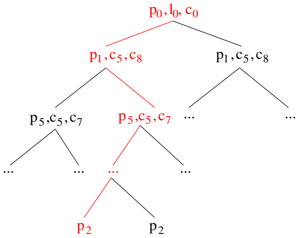




Méthode par Intersection des Lieux Géométriques (LIM)

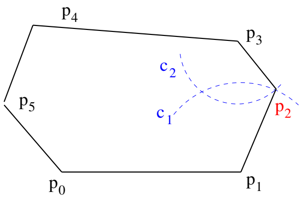
- Obtention du plan de construction
- Évaluation du plan de construction
- Arbre d'interprétation
  - Nombre exponentiel de branches en le nb d'intersections.

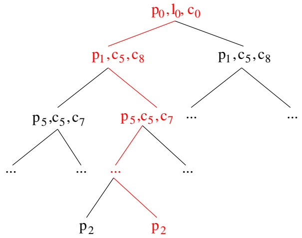




Méthode par Intersection des Lieux Géométriques (LIM)

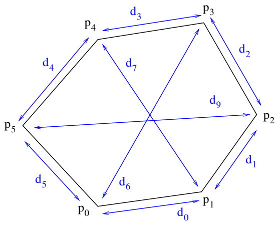
- Obtention du plan de construction
- Évaluation du plan de construction
- Arbre d'interprétation
  - Nombre exponentiel de branches en le nb d'intersections.





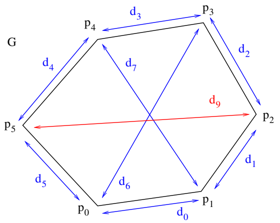
Méthode par Intersection des Lieux Géométriques (LIM)

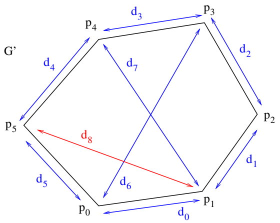
- Obtention du plan de construction
- Évaluation du plan de construction
- Arbre d'interprétation
- Échec de la méthode : hexagone $K_{3,3}$



Reparamétrisation

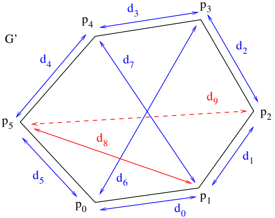
- Remplacer d contraintes d'un SCG G pour obtenir un SCG G' soluble par LIM.





Reparamétrisation

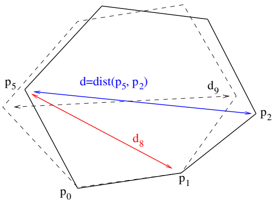
- Remplacer d contraintes d'un SCG G pour obtenir un SCG G' soluble par LIM.
- Plan de construction $Cp(k)$ paramétré par les valeurs k des contraintes ajoutées.



Parameters :
$p_0, l_0, d_0, \ldots, d_5, \ldots, d_8$.
Unknowns :
$c_0, p_1, c_1, c_2, p_5, \ldots$
Terms :
$c_0 = mkcircle(p_0, d_0)$
$p_1 = intercl(c_0, l_0)$
$c_1 = mkcircle(p_0, d_5)$
$c_2 = mkcircle(p_1, d_8)$
$p_5 = intercc(c_1, c_2)$
$\ldots$

Reparamétrisation

- Remplacer d contraintes d'un SCG G pour obtenir un SCG G' soluble par LIM.
- Plan de construction $Cp(k)$ paramétré par les valeurs k des contraintes ajoutées.
- $F(k) = 0 \Leftrightarrow$ la figure produite par $Cp(k)$ satisfait les contraintes supprimées.



Parameters :
$p_0, l_0, d_0, \ldots, d_5, \ldots, d_8.$
Unknowns :
$c_0, p_1, c_1, c_2, p_5, \ldots$
Terms :
$c_0 = mkcircle(p_0, d_0)$
$p_1 = intercl(c_0, l_0)$
$c_1 = mkcircle(p_0, d_5)$
$c_2 = mkcircle(p_1, d_8)$
$p_5 = intercc(c_1, c_2)$
$\ldots$

$F(k_0) = dist(p_5, p_2) - d_9$

Nos Apports

- Remplacer d contraintes d'un SCG G pour obtenir un SCG G' soluble par LIM.
  - Stratégie de décomposition du problème.

- $F(k) = 0 \Leftrightarrow$ la figure produite par $Cp(k)$ satisfait les contraintes supprimées.
  - Échantillonnage de F sur toutes les branches de $Cp(k)$.
  - Utilisation d'une méthode par continuation.

Nos Apports

- Remplacer d contraintes d'un SCG G pour obtenir un SCG G' soluble par LIM.
  - Stratégie de décomposition du problème.

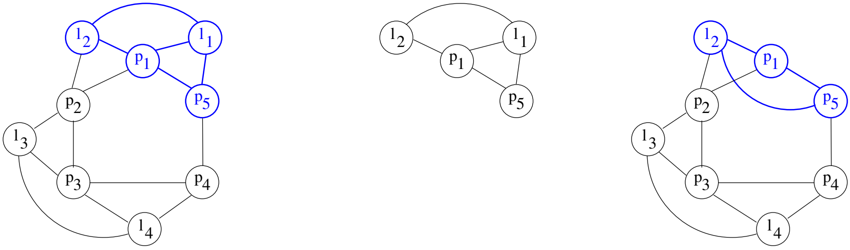
- $F(k) = 0 \Leftrightarrow$ la figure produite par $Cp(k)$ satisfait les contraintes supprimées.
  - Échantillonnage de F sur toutes les branches de $Cp(k)$.
  - Utilisation d'une méthode par continuation.

Décomposition

- SCG plus petits résolus indépendamment,
- moins de contraintes remplacées dans chaque SCG.

Principe : Soit G struct. bien contraint.

- Identifier $G_1 < G$ struct. bien contraint,
  - résoudre G_1,
  - calculer son bord dans G : $B_{G_1/G}$
- recommencer avec $G - G_1 + B_{G_1/G}$.



Reparamétrisation avec décomposition

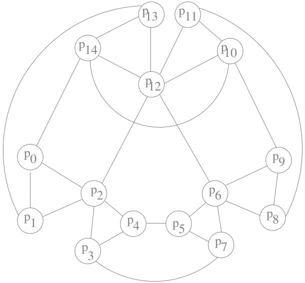
- G_c graphe partiel induit par les sommets de G construits.
- pour $s \notin G_c$,
  - $DEGC(s)$ =nombre de sommets de G_c adjacents à s.
  - $D(s) = DEGC(s) - DOF(s)$ est la constriction de s.

Procédure : `DecompEtReparam`
Entrée : SCG G bien contraint.

1 : Choisir un repère.
2 : SI $\exists v \in G \setminus G_c \mid D(v) = 0$, ajouter s à G_c.

FIN SI
SI G_c est bien contraint
ET contient un sommet intérieur,

$G' = G - G_c + B_{G_c/G}$
`DecompEtReparam`(G').
SINON aller à 2.
FIN SI



Reparamétrisation avec décomposition

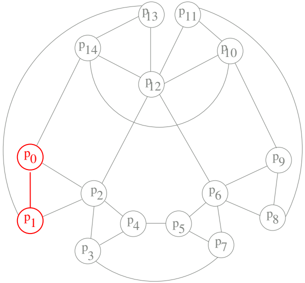
- G_c graphe partiel induit par les sommets de G construits.
- pour $s \notin G_c$,
  - $DEGC(s)$ =nombre de sommets de G_c adjacents à s.
  - $D(s) = DEGC(s) - DOF(s)$ est la constriction de s.

Procédure : `DecompEtReparam`
Entrée : SCG G bien contraint.

1 : Choisir un repère.
2 : SI $\exists v \in G \setminus G_c \mid D(v) = 0$, ajouter s à G_c.

FIN SI
SI G_c est bien contraint
ET contient un sommet intérieur,

$G' = G - G_c + B_{G_c/G}$
`DecompEtReparam`(G').
SINON aller à 2.
FIN SI



Reparamétrisation avec décomposition

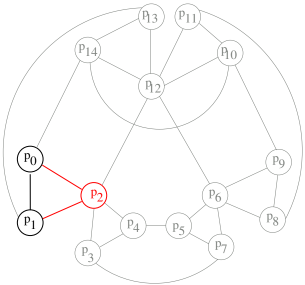
- G_c graphe partiel induit par les sommets de G construits.
- pour $s \notin G_c$,
  - $DEGC(s)$ =nombre de sommets de G_c adjacents à s.
  - $D(s) = DEGC(s) - DOF(s)$ est la constriction de s.

Procédure : `DecompEtReparam`
Entrée : SCG G bien contraint.

1 : Choisir un repère.
2 : SI $\exists v \in G \setminus G_c \mid D(v) = 0$, ajouter s à G_c.

FIN SI
SI G_c est bien contraint
ET contient un sommet intérieur,

$G' = G - G_c + B_{G_c/G}$
`DecompEtReparam`(G').
SINON aller à 2.
FIN SI



Reparamétrisation avec décomposition

- G_c graphe partiel induit par les sommets de G construits.
- pour $s \notin G_c$,
  - $DEGC(s)$ =nombre de sommets de G_c adjacents à s.
  - $D(s) = DEGC(s) - DOF(s)$ est la constriction de s.

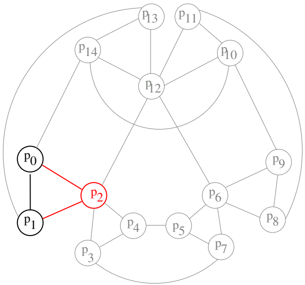
Procédure : `DecompEtReparam`
Entrée : SCG G bien contraint.

1 : Choisir un repère.
2 : SI $\exists v \in G \setminus G_c \mid D(v) = 0$, ajouter s à G_c.

FIN SI
SI G_c est bien contraint
ET contient un sommet intérieur,

$$G' = G - G_c + B_{G_c/G}$$

`DecompEtReparam`(G').
SINON aller à 2.
FIN SI



Reparamétrisation avec décomposition

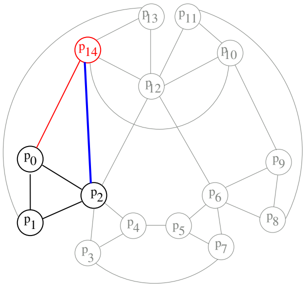
- G_c graphe partiel induit par les sommets de G construits.
- pour $s \notin G_c$,
  - $DEGC(s)$ =nombre de sommets de G_c adjacents à s.
  - $D(s) = DEGC(s) - DOF(s)$ est la constriction de s.

Procédure : `DecompEtReparam`
Entrée : SCG G bien contraint.

1 : Choisir un repère.
2 : SI $\exists v \in G \setminus G_c \mid D(v) = 0$, ajouter s à G_c.
SINON Choisir $v \in G \setminus G_c$ maximisant $D(s)$.
SI $D(s) < 0$, ajouter $D(s)$ contraintes
entre s et des sommets de G_c

Ajouter s à G_c
FIN SI
SI G_c est bien contraint
ET contient un sommet intérieur,

$G' = G - G_c + B_{G_c/G}$
`DecompEtReparam`(G').
SINON aller à 2.
FIN SI



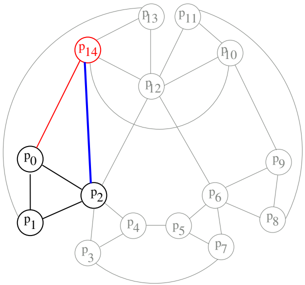
Reparamétrisation avec décomposition

- G_c graphe partiel induit par les sommets de G construits.
- pour $s \notin G_c$,
  - $DEGC(s)$ =nombre de sommets de G_c adjacents à s.
  - $D(s) = DEGC(s) - DOF(s)$ est la constriction de s.

Procédure : `DecompEtReparam`
Entrée : SCG G bien contraint.

1 : Choisir un repère.
2 : SI $\exists v \in G \setminus G_c \mid D(v) = 0$, ajouter s à G_c.
SINON Choisir $v \in G \setminus G_c$ maximisant $D(s)$.
SI $D(s) < 0$, ajouter $D(s)$ contraintes entre s et des sommets de G_c

Ajouter s à G_c
FIN SI
SI G_c est bien contraint
ET contient un sommet intérieur,
ET autant de contraintes ajoutées que supprimées,
$G' = G - G_c + B_{G_c/G}$
`DecompEtReparam(`G'`).`
SINON aller à 2.
FIN SI



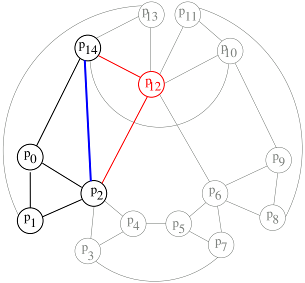
Reparamétrisation avec décomposition

- G_c graphe partiel induit par les sommets de G construits.
- pour $s \notin G_c$,
  - $DEGC(s)$ =nombre de sommets de G_c adjacents à s.
  - $D(s) = DEGC(s) - DOF(s)$ est la constriction de s.

Procédure : `DecompEtReparam`
Entrée : SCG G bien contraint.

1 : Choisir un repère.
2 : SI $\exists v \in G \setminus G_c \mid D(v) = 0$, ajouter s à G_c.
SINON Choisir $v \in G \setminus G_c$ maximisant $D(s)$.
SI $D(s) < 0$, ajouter $D(s)$ contraintes entre s et des sommets de G_c

Ajouter s à G_c
FIN SI
SI G_c est bien contraint
ET contient un sommet intérieur,
ET autant de contraintes ajoutées que supprimées,
$G' = G - G_c + B_{G_c/G}$
`DecompEtReparam`(G').
SINON aller à 2.
FIN SI



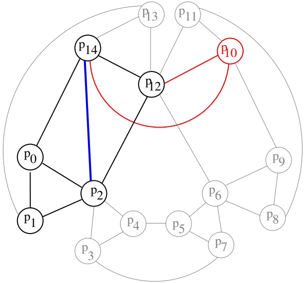
Reparamétrisation avec décomposition

- G_c graphe partiel induit par les sommets de G construits.
- pour $s \notin G_c$,
  - $DEGC(s)$ =nombre de sommets de G_c adjacents à s.
  - $D(s) = DEGC(s) - DOF(s)$ est la constriction de s.

Procédure : `DecompEtReparam`
Entrée : SCG G bien contraint.

1 : Choisir un repère.
2 : SI $\exists v \in G \setminus G_c \mid D(v) = 0$, ajouter s à G_c.
SINON Choisir $v \in G \setminus G_c$ maximisant $D(s)$.
SI $D(s) < 0$, ajouter $D(s)$ contraintes
entre s et des sommets de G_c

Ajouter s à G_c
FIN SI
SI G_c est bien contraint
ET contient un sommet intérieur,
ET autant de contraintes ajoutées que supprimées,
$G' = G - G_c + B_{G_c/G}$
`DecompEtReparam`(G').
SINON aller à 2.
FIN SI



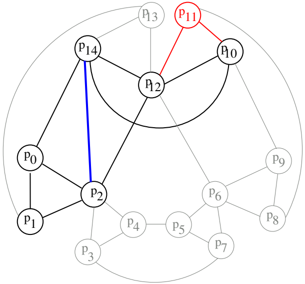
Reparamétrisation avec décomposition

- G_c graphe partiel induit par les sommets de G construits.
- pour $s \notin G_c$,
  - $DEGC(s)$ =nombre de sommets de G_c adjacents à s.
  - $D(s) = DEGC(s) - DOF(s)$ est la constriction de s.

Procédure : `DecompEtReparam`
Entrée : SCG G bien contraint.

1 : Choisir un repère.
2 : SI $\exists v \in G \setminus G_c \mid D(v) = 0$, ajouter s à G_c.
SINON Choisir $v \in G \setminus G_c$ maximisant $D(s)$.
SI $D(s) < 0$, ajouter $D(s)$ contraintes entre s et des sommets de G_c

Ajouter s à G_c
FIN SI
SI G_c est bien contraint
ET contient un sommet intérieur,
ET autant de contraintes ajoutées que supprimées,
$G' = G - G_c + B_{G_c/G}$
`DecompEtReparam`(G').
SINON aller à 2.
FIN SI

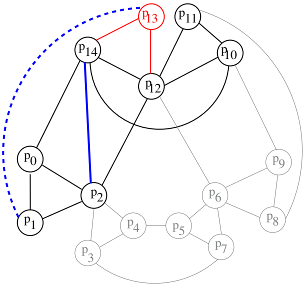


Reparamétrisation avec décomposition

- G_c graphe partiel induit par les sommets de G construits.
- pour $s \notin G_c$,
  - $DEGC(s)$ =nombre de sommets de G_c adjacents à s.
  - $D(s) = DEGC(s) - DOF(s)$ est la constriction de s.

Procédure : `DecompEtReparam`
Entrée : SCG G bien contraint.

1 : Choisir un repère.
2 : SI $\exists v \in G \setminus G_c \mid D(v) = 0$, ajouter s à G_c.
SINON Choisir $v \in G \setminus G_c$ maximisant $D(s)$.
SI $D(s) < 0$, ajouter $D(s)$ contraintes entre s et des sommets de G_c
SI $D(s) > 0$, enlever $D(s)$ contraintes entre s et des sommets de G_c
Ajouter s à G_c
FIN SI
SI G_c est bien contraint
ET contient un sommet intérieur,
ET autant de contraintes ajoutées que supprimées,
$G' = G - G_c + B_{G_c/G}$
`DecompEtReparam`(G').
SINON aller à 2.
FIN SI

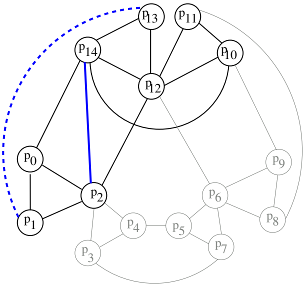


Reparamétrisation avec décomposition

- G_c graphe partiel induit par les sommets de G construits.
- pour $s \notin G_c$,
  - $DEGC(s)$ =nombre de sommets de G_c adjacents à s.
  - $D(s) = DEGC(s) - DOF(s)$ est la constriction de s.

Procédure : `DecompEtReparam`
Entrée : SCG G bien contraint.

1 : Choisir un repère.
2 : SI $\exists v \in G \setminus G_c \mid D(v) = 0$, ajouter s à G_c.
SINON Choisir $v \in G \setminus G_c$ maximisant $D(s)$.
SI $D(s) < 0$, ajouter $D(s)$ contraintes entre s et des sommets de G_c
SI $D(s) > 0$, enlever $D(s)$ contraintes entre s et des sommets de G_c
Ajouter s à G_c
FIN SI
SI G_c est bien contraint
ET contient un sommet intérieur,
ET autant de contraintes ajoutées que supprimées,
$G' = G - G_c + B_{G_c/G}$
`DecompEtReparam`(G').
SINON aller à 2.
FIN SI

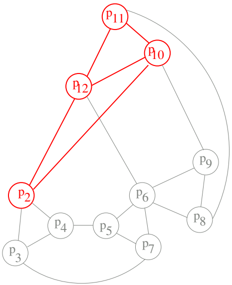


Reparamétrisation avec décomposition

- G_c graphe partiel induit par les sommets de G construits.
- pour $s \notin G_c$,
  - $DEGC(s)$ =nombre de sommets de G_c adjacents à s.
  - $D(s) = DEGC(s) - DOF(s)$ est la constriction de s.

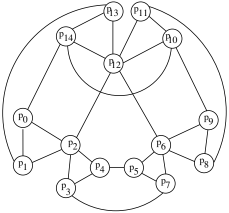
Procédure : `DecompEtReparam`
Entrée : SCG G bien contraint.

1 : Choisir un repère.
2 : SI $\exists v \in G \setminus G_c \mid D(v) = 0$, ajouter s à G_c.
SINON Choisir $v \in G \setminus G_c$ maximisant $D(s)$.
SI $D(s) < 0$, ajouter $D(s)$ contraintes entre s et des sommets de G_c
SI $D(s) > 0$, enlever $D(s)$ contraintes entre s et des sommets de G_c
Ajouter s à G_c
FIN SI
SI G_c est bien contraint
ET contient un sommet intérieur,
ET autant de contraintes ajoutées que supprimées,
$G' = G - G_c + B_{G_c/G}$
DecompEtReparam(G').
SINON aller à 2.
FIN SI

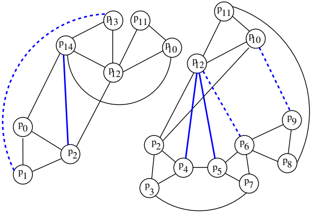


Résultats

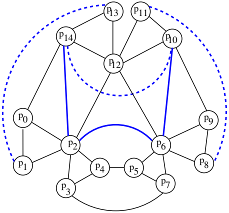
SCG original :



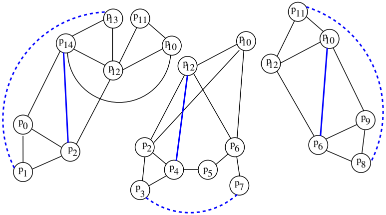
Reparamétrisation & décomposition : N=2



Reparamétrisation globale : N=3



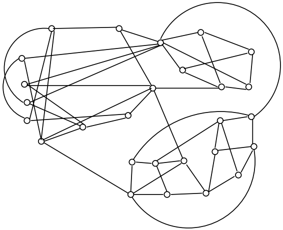
Heuristique : N=1



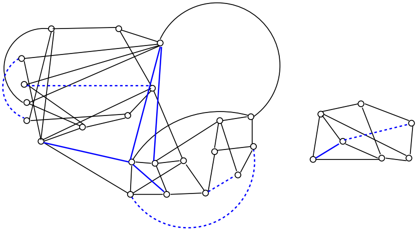
N : nombre maximum de contraintes rattrapées par sous-système

Résultats

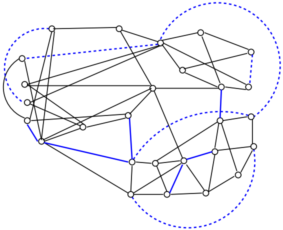
SCG original :



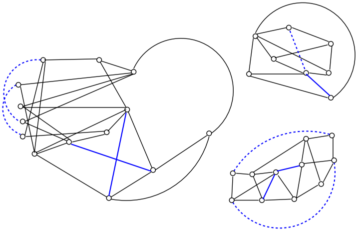
Reparamétrisation & décomposition : N=4



Reparamétrisation globale :
N=6



Heuristique : N=2



N : nombre maximum de contraintes rattrapées par sous-système

Résolution numérique

- Plan de construction Pc

Paramètres :
$h_0, h_1, \ldots, h_i, h_{i+1}, \ldots h_{m-2}, h_{m-1}$
Inconnues :
$c_0, c_1, \ldots, p_0, p_1, \ldots, l_0, l_1, \ldots$
Termes :
...
$c_1 = mkcircle(p_0, h_1)$
$c_2 = mkcircle(p_1, h_2)$
$p_2 = intercc(c_1, c_2)$
...
$l_i = mkline(p_{i_0}, p_{i_1}, h_i)$
$c_i = mkcircle(p_{i_2}, k_i)$
$p_i = intercl(c_i, l_i)$
...
$f_0 = eval\_dist(p_{j_0}, p_{j_1}) - h_j$
$f_1 = eval\_angle(l_{j_2}, l_{j_3}) - h_{j+1}$
...

Résolution numérique

- Plan de construction $Pc(k, \ )$
- Contraintes ajoutées : paramètres guides $k = (k_0, ..., k_{d-1})$

Paramètres :
$h_0, h_1, \ldots, h_i, h_{i+1}, \ldots h_{m-2}, h_{m-1}$
Inconnues :
$c_0, c_1, \ldots, p_0, p_1, \ldots, l_0, l_1, \ldots$
Termes :
$\ldots$
$c_1 = mkcircle(p_0, h_1)$
$c_2 = mkcircle(p_1, h_2)$
$p_2 = intercc(c_1, c_2)$
$\ldots$
$l_i = mkline(p_{i_0}, p_{i_1}, h_i)$
$c_i = mkcircle(p_{i_2}, k_i)$
$p_i = intercl(c_i, l_i)$
$\ldots$
$f_0 = eval\_dist(p_{j_0}, p_{j_1}) - h_j$
$f_1 = eval\_angle(l_{j_2}, l_{j_3}) - h_{j+1}$
$\ldots$

Résolution numérique

- Plan de construction $Pc(k, h)$
- Contraintes ajoutées : paramètres guides $k = (k_0, ..., k_{d-1})$
- Autres contraintes : paramètres $h = (h_0, ..., h_i, ...$ $)$

Paramètres :
$h_0, h_1, \ldots, h_i, h_{i+1}, \ldots h_{m-2}, h_{m-1}$
Inconnues :
$c_0, c_1, \ldots, p_0, p_1, \ldots, l_0, l_1, \ldots$
Termes :
$\ldots$
$c_1 = mkcircle(p_0, h_1)$
$c_2 = mkcircle(p_1, h_2)$
$p_2 = intercc(c_1, c_2)$
$\ldots$
$l_i = mkline(p_{i_0}, p_{i_1}, h_i)$
$c_i = mkcircle(p_{i_2}, k_i)$
$p_i = intercl(c_i, l_i)$
$\ldots$
$f_0 = eval\_dist(p_{j_0}, p_{j_1}) - h_j$
$f_1 = eval\_angle(l_{j_2}, l_{j_3}) - h_{j+1}$
$\ldots$

Résolution numérique

- Plan de construction $Pc(k, h)$
- Contraintes ajoutées : paramètres guides $k = (k_0, ..., k_{d-1})$
- Autres contraintes : paramètres $h = (h_0, ..., h_i, ...h_j, ...h_{m-1})$
- Évaluation des contraintes supprimées :
  $\Rightarrow$ construction de la multi-fonction F

Paramètres :
$h_0, h_1, \ldots, h_i, h_{i+1}, \ldots h_{m-2}, h_{m-1}$
Inconnues :
$c_0, c_1, \ldots, p_0, p_1, \ldots, l_0, l_1, \ldots$
Termes :
$\ldots$
$c_1 = mkcircle(p_0, h_1)$
$c_2 = mkcircle(p_1, h_2)$
$p_2 = intercc(c_1, c_2)$
$\ldots$
$l_i = mkline(p_{i_0}, p_{i_1}, h_i)$
$c_i = mkcircle(p_{i_2}, k_i)$
$p_i = intercl(c_i, l_i)$
$\ldots$
$f_0 = eval\_dist(p_{j_0}, p_{j_1}) - h_j$
$f_1 = eval\_angle(l_{j_2}, l_{j_3}) - h_{j+1}$
$\ldots$

$$
\begin{aligned}
F : \mathbb{R}^d \times \mathbb{R}^m &\to \mathbb{R}^d \\
(k, h) &\mapsto \text{choisir une branche de } Pc, \\
&\quad\ \text{évaluer } Pc(k, h), \\
&\quad\ \text{évaluer } (f_0, f_1, ..., f_{d-1}).
\end{aligned}
$$

Résolution numérique

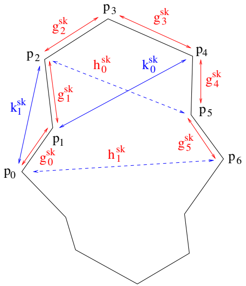
- Plan de construction Pc
- Contraintes ajoutées : paramètres guides $k = (k_0, ..., k_{d-1})$
- Autres contraintes : paramètres $h = (h_0, ..., h_i, ... \qquad)$
- Évaluation des contraintes supprimées :
  $\Rightarrow$ construction de la multi-fonction F
- F analytique

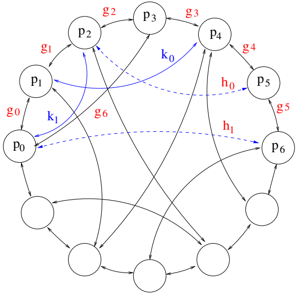
$$\begin{aligned} F : \mathbb{R}^d \times \mathbb{R}^m &\to \mathbb{R}^d \\ (k, h) &\mapsto \text{choisir une branche de } Pc, \\ &\quad\ \text{évaluer } Pc(k, h), \\ &\quad\ \text{évaluer } (f_0, f_1, ..., f_{d-1}). \end{aligned}$$

Méthode par continuation des paramètres

L'esquisse fournit une solution :
$F(k^{sk}, h^{sk}) = 0$

Solutions k cherchées :
$F(k, h^{*}) = 0$





Méthode par continuation des paramètres

L'esquisse fournit une solution :

$$F(k^{sk},h^{sk}) = 0$$

Solutions k cherchées :

$$F(k,h^{*}) = 0$$

Fonction d'homotopie :

$$H : \mathbb{R}^{d} \times \mathbb{R} \to \mathbb{R}^{d}$$

$$H(k,t) = F(k,(1-t)h^{sk} + th^{*})$$

Méthode par continuation des paramètres

L'esquisse fournit une solution :

$$F(k^{sk},h^{sk}) = 0$$

Solutions k cherchées :

$$F(k,h^*) = 0$$

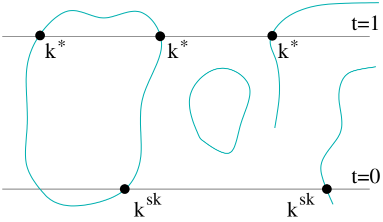
Fonction d'homotopie :

$$H : \mathbb{R}^d \times \mathbb{R} \to \mathbb{R}^d$$

$$H(k,t) = F(k,(1-t)h^{sk} + th^*)$$

Chemins d'homotopie : solutions de $H(k,t) = 0$

- variétés de dimension 1 plongées dans $\mathbb{R}^d \times \mathbb{R}$
- difféomorphes à des segments ou des cercles
- solutions à l'infini
- sortie du domaine de définition



Méthode par continuation des paramètres

L'esquisse fournit une solution :

$$F(k^{sk},h^{sk}) = 0$$

Solutions k cherchées :

$$F(k,h^{*}) = 0$$

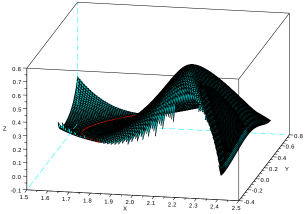
Fonction d'homotopie :

$$H : \mathbb{R}^d \times \mathbb{R} \rightarrow \mathbb{R}^d$$

$$H(k,t) = F(k,(1-t)h^{sk} + th^{*})$$

Chemins d'homotopie : solutions de $H(k,t) = 0$

- variétés de dimension 1 plongées dans $\mathbb{R}^d \times \mathbb{R}$
- difféomorphes à des segments ou des cercles
- solutions à l'infini
- sortie du domaine de définition

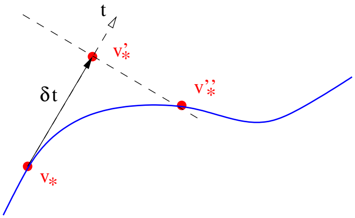


Suivi des chemins d'homotopie

$$F : \mathbb{R}^{d+1} \rightarrow \mathbb{R}^d$$
$$c = \{v \in \mathbb{R}^{d+1} | F(v) = 0\},\ v_* \in c$$

Comment suivre c depuis v_* ?

- Prédiction-correction
  - Prédire v'_* le long de la tangente t de c en v_* :
    $v'_* = v_* + \delta t, \delta \in \mathbb{R}$.
  - Corriger v'_* pour obtenir $v''_* \in c$ grâce à Newton-Raphson en ajoutant une équation qui garanti la progression le long de c.

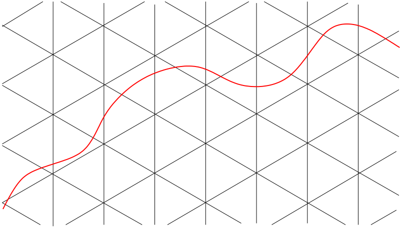


Suivi des chemins d'homotopie

$$F : \mathbb{R}^{d+1} \to \mathbb{R}^d$$
$$c = \{v \in \mathbb{R}^{d+1} | F(v) = 0\},\ v_* \in c$$

Comment suivre c depuis v_* ?

- Prédiction-correction
- Approximation linéaire par morceaux (PLI).
  - $\mathcal{M}$: maillage de $\mathbb{R}^{d+1}$ en simplexes de dimension $d+1$.
  - $\mathcal{F}$: ensemble des faces (simplexes de dimension d) de $\mathcal{M}$.

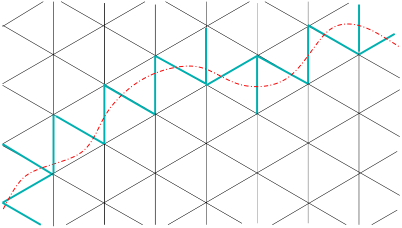


Suivi des chemins d'homotopie

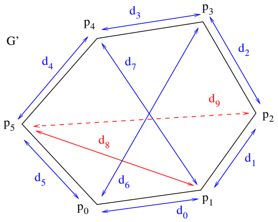
$$F : \mathbb{R}^{d+1} \to \mathbb{R}^d$$
$$c = \{v \in \mathbb{R}^{d+1} | F(v) = 0\},\ v_* \in c$$

Comment suivre c depuis v_* ?

- Prédiction-correction
- Approximation linéaire par morceaux (PLI).
  - $\mathcal{M}$: maillage de $\mathbb{R}^{d+1}$ en simplexes de dimension $d+1$.
  - $\mathcal{F}$: ensemble des faces (simplexes de dimension d) de $\mathcal{M}$.
  - $\mathcal{C} \subset \mathcal{F}$: faces dont au moins un point x satisfait $F(x) = 0$.



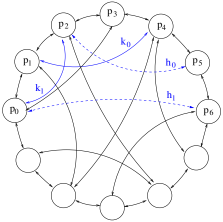
Résultats



	Sampling	Tracking
$K_{3,3}$ 6 primitives 9 constraints 1 added/removed	2 solutions, $\simeq 0.02sec$	2 solutions, $\simeq 0.4sec$
Dodecagone 12 primitives 21 constraints 2 added/removed	2 solutions, $\simeq 10sec$	2 solutions, $\simeq 1.5sec$
Icosahèdre 12 primitives 30 constraints 3 added/removed	1 solution, $\simeq 767sec$	2 solutions, $\simeq 17.4sec$

PC with processor at 2.2GHz

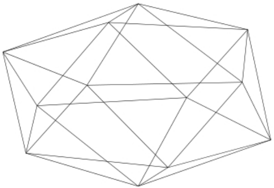
Résultats



	Sampling	Tracking
$K_{3,3}$ 6 primitives 9 constraints 1 added/removed	2 solutions, $\simeq 0.02 sec$	2 solutions, $\simeq 0.4 sec$
Dodecagone 12 primitives 21 constraints 2 added/removed	2 solutions, $\simeq 10 sec$	2 solutions, $\simeq 1.5 sec$
Icosahèdre 12 primitives 30 constraints 3 added/removed	1 solution, $\simeq 767 sec$	2 solutions, $\simeq 17.4 sec$

PC with processor at 2.2GHz

Résultats



	Sampling	Tracking
$K_{3,3}$ 6 primitives 9 constraints 1 added/removed	2 solutions, $\simeq 0.02 sec$	2 solutions, $\simeq 0.4 sec$
Dodecagone 12 primitives 21 constraints 2 added/removed	2 solutions, $\simeq 10 sec$	2 solutions, $\simeq 1.5 sec$
Icosahèdre 12 primitives 30 constraints 3 added/removed	1 solution, $\simeq 767 sec$	2 solutions, $\simeq 17.4 sec$

PC with processor at 2.2GHz

Conclusion

Résumé : G non résoluble par LIM

- Phase symbolique : sous-systèmes reparamétrés.
- Phase numérique : chaque sous-système résolu par continuation.

Perspectives :

- Résolution du suivi des chemins d'homotopie.
- Trouver d'autres chemins que celui de l'esquisse.
- Adapter la reparamétrisation à d'autres décompositions.