Isolation de bâtiments : un processus et un logiciel pour la conception de panneaux , projet ISOBIM

Michel Aldanondo, Julien Lesbegueries, Andrea Christophe, Élise Vareilles, Xavier Lorca

CIGI QUALITA MOSIM 2023
Isolation de bâtiments : un processus et un logiciel pour la conception de panneaux, projet ISOBIM

MICHEL ALDANONDO[1], JULIEN LESBEGUERIES[1], ANDREA CHRISTOPHE[1,2], ELISE VAREILLES[3], XAVIER LORCA[1]

[1] Université de Toulouse / IMT Mines Albi / Centre Génie Industriel – Albi, France
2 Armines – Paris, France
3 Université de Toulouse / ISAE-SUPAERO / DISC – Toulouse, France
firstame.lastname@mines-albi.fr, firstame.lastname@isae-supaero.fr

***Résumé* – Dans le cadre de la rénovation énergétique de bâtiments, l'objectif de cette communication est de montrer comment la conception incluant le calepinage de panneaux d'isolation peut être fortement assisté par des outils logiciels. Ce travail a été réalisé dans le cadre d'un projet de recherche collaborative appliquée (ISOBIM). Les bâtiments considérés sont des petits collectifs de dix étages maximum construits entre 1945 et 1980. La solution technique d'isolation retenue exploite le principe de panneaux préfabriqués comprenant portes et fenêtres. Les résultats proposés rassemblent : (i) un processus de conception incluant un calepinage, (ii) différents modèles de façade supportant ce processus et (iii) un algorithme de conception ou calepinage des panneaux. Un exemple illustre les propositions et permet de les discuter.**

***Abstract* – Within the framework of the energy building renovation, the objective of this communication is to show how the design including the layout of the insulation panels can be supported by software tools. This work was carried during an applied collaborative research project (ISOBIM). The buildings considered are small collective buildings with a maximum of ten floors built between 1945 and 1980. The technical solution of insulation works with the principle of prefabricated panels including doors and windows. The proposed results include: (i) a process for designing the layout, (ii) different façade models and (iii) an algorithm for designing the panels layout. An example illustrates the proposals and discusses them.**

***Mots clés* - Rénovation et isolation de bâtiment, Conception et calepinage de panneaux, Système d'aide à la conception, Problème de satisfaction de contraintes.**

***Keywords* - Building insulation renovation, Panel layout design, Aiding design system, Constraint-based problem**

1 INTRODUCTION

Selon la Commission européenne [European Commission, 2020], "Aujourd'hui, environ 75 % du parc immobilier de l'UE est inefficace sur le plan énergétique. Cela signifie qu'une grande partie de l'énergie utilisée est gaspillée. Ces pertes d'énergie peuvent être minimisées en améliorant les bâtiments existants". Le même rapport souligne que la rénovation des bâtiments peut réduire la consommation totale d'énergie de l'UE de 5 à 6 % et les émissions de dioxyde de carbone d'environ 5 %.

En conséquence, de nombreuses agences de recherche européennes et nationales ont lancé des programmes de recherche pour encourager la rénovation systématique des bâtiments. En France, l'agence nationale de la recherche (ANR) soutient un projet collaboratif appelé ISOBIM. L'objectif de ce projet est de développer des outils logiciels d'aide à la décision libres, intégrés et disponibles sur le web pour soutenir les acteurs du processus de rénovation utilisant des panneaux modulaires préfabriqués à ossature bois. Dans ce projet, nous considérons les panneaux de rénovation "lourds", c'est-à-dire qui intègrent de nouvelles portes et fenêtres, par opposition aux solutions "légères" qui contournent les anciennes. Le lecteur peut consulter [Urbikain, 2020] ou [International Energy Agency, 2011] pour plus de détails.

Il est important de noter que les "solutions légères" nécessitent un travail important de dimensionnement, de découpe et d'ajustement des panneaux, qui doit être effectué sur le chantier pendant la rénovation. En revanche, les "solutions lourdes" supposent que tous les panneaux soient préfabriqués en usine, pour être ensuite assemblés sans retouches sur le chantier. Les "solutions légères" sont plus souvent liées à une approche artisanale tandis que les "solutions lourdes" correspondent davantage à une activité industrielle. Les solutions proposées dans cet article s'inscrivent pleinement dans une démarche industrielle qui permet de réduire les coûts de fabrication et de montage ainsi que la durée de la rénovation sur site.

Pour fixer des idées sur les "solutions lourdes", quelques ordres de grandeur des panneaux utilisés dans les "solutions lourdes" sont donnés. En termes de dimensions, la taille maximale des panneaux est d'environ 3,5 mètres par 13 mètres en raison des contraintes de transport. En termes de poids, un tel panneau (3,5x13) peut peser entre 1,5 et 4 tonnes. En ce qui concerne le coût, y compris la fabrication et l'installation, un panneau sans porte ni fenêtre coûte entre 150 et 250 €/m2 en Europe. En ce qui concerne l'amélioration de la performance énergétique, il est difficile de fournir un résultat exact, mais une telle solution ne peut être utilisée que si les coûts de chauffage sont au minimum divisés par 2 ou 3.

Quelques modèles de processus de rénovation globale peuvent être trouvés dans la littérature. Pour notre travail, nous considérons celui publié dans [Aldanondo et al, 2014] qui considère quatre étapes principales : (i) numérisation du bâtiment, (ii) conception et calepinage des panneaux, (iii) fabrication des panneaux (iv) montage des panneaux sur le site (figure 1). Cette communication traite de la deuxième étape

(conception de la disposition des panneaux) et de la transition entre la première et la deuxième étape (préparation de la conception juste après la numérisation du bâtiment).

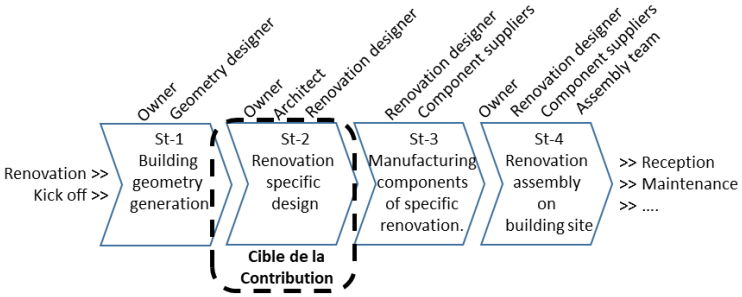

Figure 1. Situation des travaux

Les travaux de recherche et la présente communication suivent une approche progressive de résolution des problèmes où trois questions sont successivement abordées. Tout d'abord, les tâches du processus de conception complet sont définies. Ensuite, les données alimentant ces tâches et résultant de celles-ci sont identifiées et modélisées. Enfin, la tâche centrale de la conception est instrumentée avec un algorithme de recherche de solutions. En conséquence, la contribution est organisée comme suit. Dans une première section, nous discuterons et proposerons un modèle du processus de conception qui commence après la numérisation du bâtiment et se termine après la conception finale du calepinage des panneaux, représenté comme la "cible de la contribution" dans la figure 1. Dans une deuxième section, nous discuterons et proposerons trois modèles de façade de bâtiment qui supportent les tâches du processus précédent. Dans une troisième partie, nous discuterons et proposerons un algorithme qui peut automatiser la conception du calepinage des panneaux.

2 PROCESSUS DE CALEPINAGE

L'objectif de cette section est de clarifier et détailler les tâches nécessaires à l'ensemble du processus de conception des panneaux. Nous détaillerons d'abord la nécessité d'une étape de conception manuelle, puis la nécessité d'une étape de préparation avant la conception manuelle. Cela nous permettra de conclure avec le processus global auquel nous ajouterons une étape de correction finale.

2.1 Etape de conception manuelle

Selon certains auteurs, comme [Hillebrand et al 2014] ou [Barco et al 2014], il apparaît clairement que les algorithmes capables de résoudre de manière autonome le problème de calepinage de panneaux nécessitent des hypothèses fortes concernant les entités géométriques mises en œuvre.

Nous suivons ces idées et limitons notre algorithme de calepinage automatique aux façades de bâtiment rectangulaires en exploitant des panneaux de forme rectangulaire. Par conséquent, les façades avec des pignons triangulaires ou des formes non conventionnelles nécessiteront des panneaux non standards dont la conception sera manuelle.

Nous supposons également que les panneaux doivent être principalement supportés et fixés par leur partie basse (principalement pour supporter le poids) et fixé plus légèrement sur leur partie haute (principalement pour le contreventement). Les zones de la façade permettant la fixation en haut et en bas des panneaux, correspondent le plus souvent aux nez de dalles et aux murs de refends. Par conséquent, des parties spécifiques de la façade comme les acrotères (haut de la façade) ou les sous-bassement (bas de la façade) qui n'ont pas de zone de résistance en haut ou en bas nécessiteront également des panneaux non standards et une conception manuelle.

Ces panneaux conçus et disposés manuellement sur la façade seront à prendre en compte par le calepinage automatique comme des panneaux déjà existants devant être contournés sans espace. Les panneaux positionnés automatiquement devront leurs être contigus sans laisser le moindre vide.

2.2 Etape de préparation manuelle avant la conception

Si les façades des bâtiments ont des fenêtres et des portes qui seront remplacées et intégrées lors de la fabrication des panneaux, elles peuvent aussi avoir de multiples singularités comme : des blocs de climatisation, des panneaux solaires, des balcons, des lumières, des portes de garage, des tuyaux de descente d'eau... Pour toutes ces singularités, il faut décider lors de la tâche de préparation, si elles doivent être associées à un trou dans le panneau ou si elles doivent être contournées par les panneaux (de manière similaire aux panneaux précédemment calepinés manuellement). Les éléments géométriques rectangulaires rassemblant : (i) les singularités associées à des trous dans le panneau, (ii) les fenêtres et (iii) les portes seront appelés à partir de maintenant "*outin*". Les éléments rassemblant les singularités contournées par les panneaux, ainsi que les panneaux calepinés manuellement seront appelés à partir de maintenant "*outout*".

Comme nous l'avons expliqué précédemment, les panneaux sont fixés en haut et en bas par des supports mécaniques. Ces supports sont scellés dans la façade à des endroits spécifiques appelés zones de résistance de la façade qui peuvent supporter des efforts importants, notons qu'un grand panneau isolant (12 mètres par 3 mètres) peut peser jusqu'à 3 tonnes. Lors de la préparation, ces zones spécifiques (nez de dalle et murs de refend le plus souvent) doivent être identifiées sur la géométrie du bâtiment par des experts du bâtiment De plus, pour l'algorithme de calepinage automatique, elles doivent être approximées par des lignes de support où les supports de panneaux seront scellés. Ces lignes de support dénommées "*support line*" peuvent être verticales ou horizontales.

Au bilan la préparation débouche sur une géométrie mettant en jeux trois type d'entités, les *support line*, les *outin* à intégrer dans les panneaux et les *outout* devant être contournés par les panneaux.

2.3 Processus complet de calepinage

Compte tenu des hypothèses précédentes, nous proposons le processus de calepinage de panneaux en quatre étapes décrit en figure 2. Les entrées de ce processus correspondent à la géométrie "brute" de la façade et aux paramètres et contraintes de la solution technique d'isolation. La sortie du processus est une nomenclature de panneaux dimensionnés et positionnés.

La première étape, "Préparation", identifie et transforme les zones de résistance en *support line* et associe tout élément de façade soit à un *outin* (qui sera intégré dans un seul panneau) soit à un *outout* (qui sera contourné par les panneaux). La deuxième étape, "Conception manuelle", permet à l'utilisateur de concevoir manuellement les panneaux qui ne peuvent pas être pris en compte par les algorithmes automatiques, tels que : les panneaux non rectangulaires, les panneaux pour acrotère ou sous-bassement. La troisième étape de "Calepinage automatique" sera détaillée dans la section suivante. Une dernière étape, "Retouche manuelle", permet à l'architecte ou au propriétaire du bâtiment de corriger manuellement la disposition des panneaux proposée.

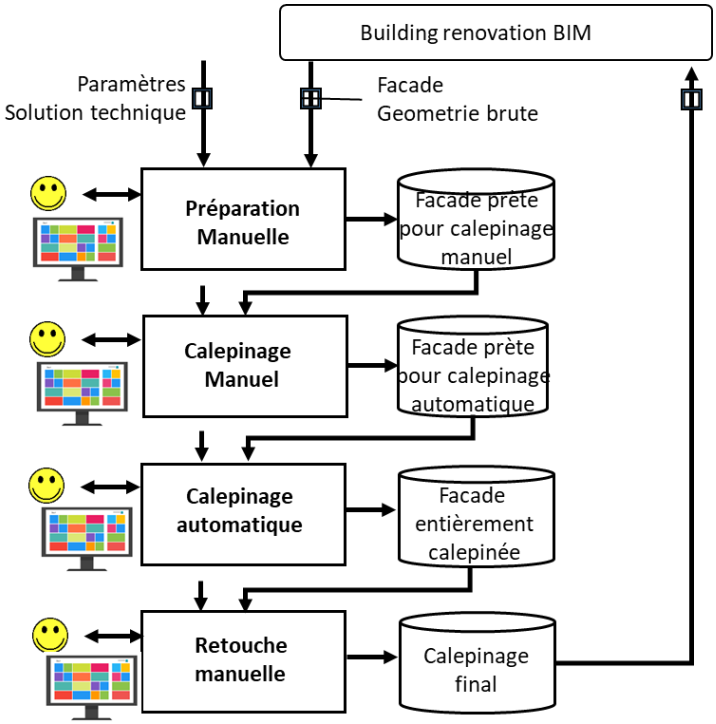

Figure 2. Processus de calepinage

3 LES MODELES DE LA FAÇADE SUPPORTANT LE PROCESSUS

Compte tenu du processus précédent, nous décrivons dans cette section les trois modèles de façade qui sont utilisés pendant le calepinage des panneaux.

Le premier modèle (figure 3) est la première description ou géométrie brute des façades du bâtiment avec des éléments caractérisés comme suit. Le bâtiment est un ensemble de façades, où chaque façade regroupe quatre types d'éléments : non résistant, résistant, singularité et additionnel. Les éléments non résistants correspondent aux portes et aux fenêtres, les éléments résistants aux nez de dalle et murs de refend. Les singularités regroupent les blocs climatisation, balcons, portes de garage, lumières et tout ce qui sera par la suite associé à un trou dans un panneau (*outin*) ou contourné par les panneaux (*outout*). Enfin des éléments additionnels peuvent compléter la façade rectangulaire et regroupent acrotères, sous-bassement et pignons triangulaires.

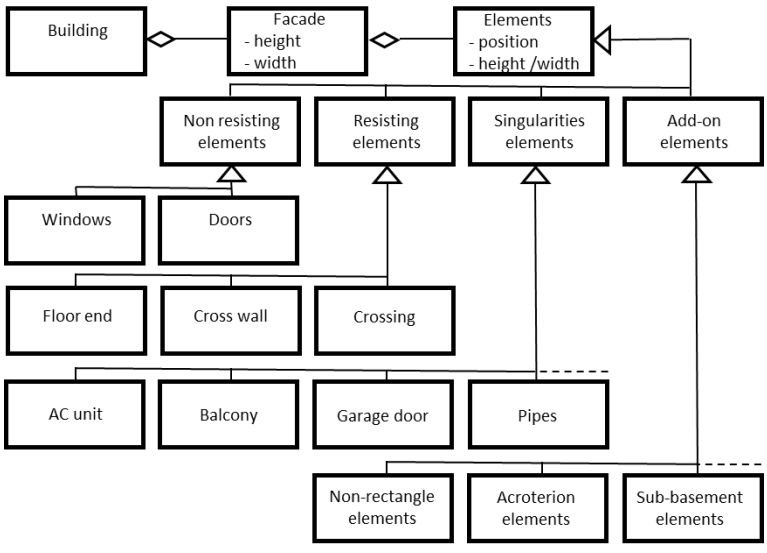

Figure 3. Modèle de base des façades

Le deuxième modèle (partie supérieure de la figure 4) est le résultat de la préparation et correspond à la façade prête pour le calepinage manuel. Ce modèle ne rassemble que trois types d'éléments : Les lignes de fixation *support line*, les éléments *outin* (qui doivent être intégrés dans un seul panneau) et les éléments *outout* (qui doivent être contournés par les panneaux). Le dernier modèle (partie inférieure de la figure 4) montre la solution de calepinage comme un ensemble de panneaux dimensionnés et positionnés. Chaque panneau comprend les éléments *outin* et des supports mécaniques.

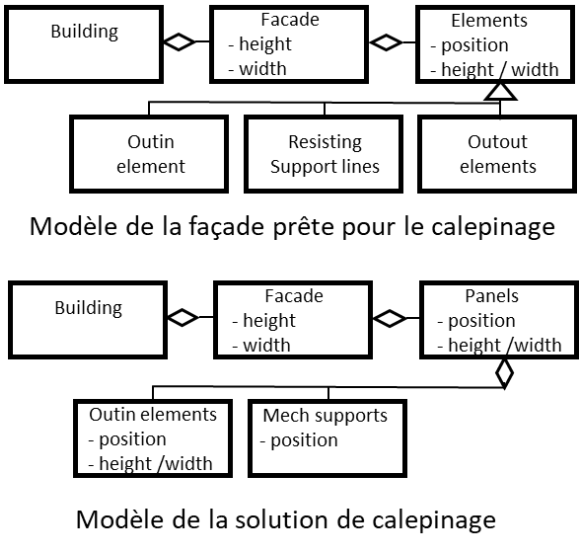

Figure 4. Modèle de la façade avant et après calepinage

4 LA SOLUTION DE CALEPINAGE AUTOMATIQUE

Dans un premier temps, nous rappelons les données du problème, les paramètres et contraintes technologiques des panneaux et les critères d'optimisation. Ensuite, nous proposons un petit état de l'art du problème pour introduire et justifier notre approche heuristique. Enfin, nous présentons les idées principales de notre algorithme de calepinage automatique de panneaux d'isolation.

4.1 Entrées, technologies et critères d'optimisation

Compte tenu des éléments précédemment décrits, l'algorithme de calepinage automatique ne considère en entrée que les trois éléments suivants : *support line*, *outin* et *outout*, dont un exemple est donné en figure 5. Cet exemple ne nécessite pas de conception manuelle des panneaux car tout est rectangulaire et, de plus, il n'y a pas de panneau additionnel pour le haut et le bas de la façade. La plupart des fabricants de panneaux recommandent d'associer à chaque couple d'éléments (balcon, fenêtre) un seul rectangle *outin* et de faire un trou dans un unique panneau.

En termes de technologie d'isolation, un panneau a des dimensions minimales et maximales (largeur et hauteur). Pour des contraintes de transport, chaque dimension doit être inférieure à la longueur du camion (environ 14 mètres) et une dimension doit être inférieure à la hauteur du camion (environ 3,5 mètres). La solution de calepinage, est un ensemble de panneaux qui couvre toute la façade sauf les *outout* qui sont contournés. Les dimensions des panneaux doivent être supérieures à celles des éléments intégrés de type *outin*. Pour la rigidité du panneau, une distance minimale entre le bord de tout *outin* et le bord du panneau existe. Chaque panneau a un poids globalement proportionnel à sa surface. Le bord inférieur du panneau peut être soutenu, suivant sa masse, soit par un ensemble de fixations sur une support line horizontale

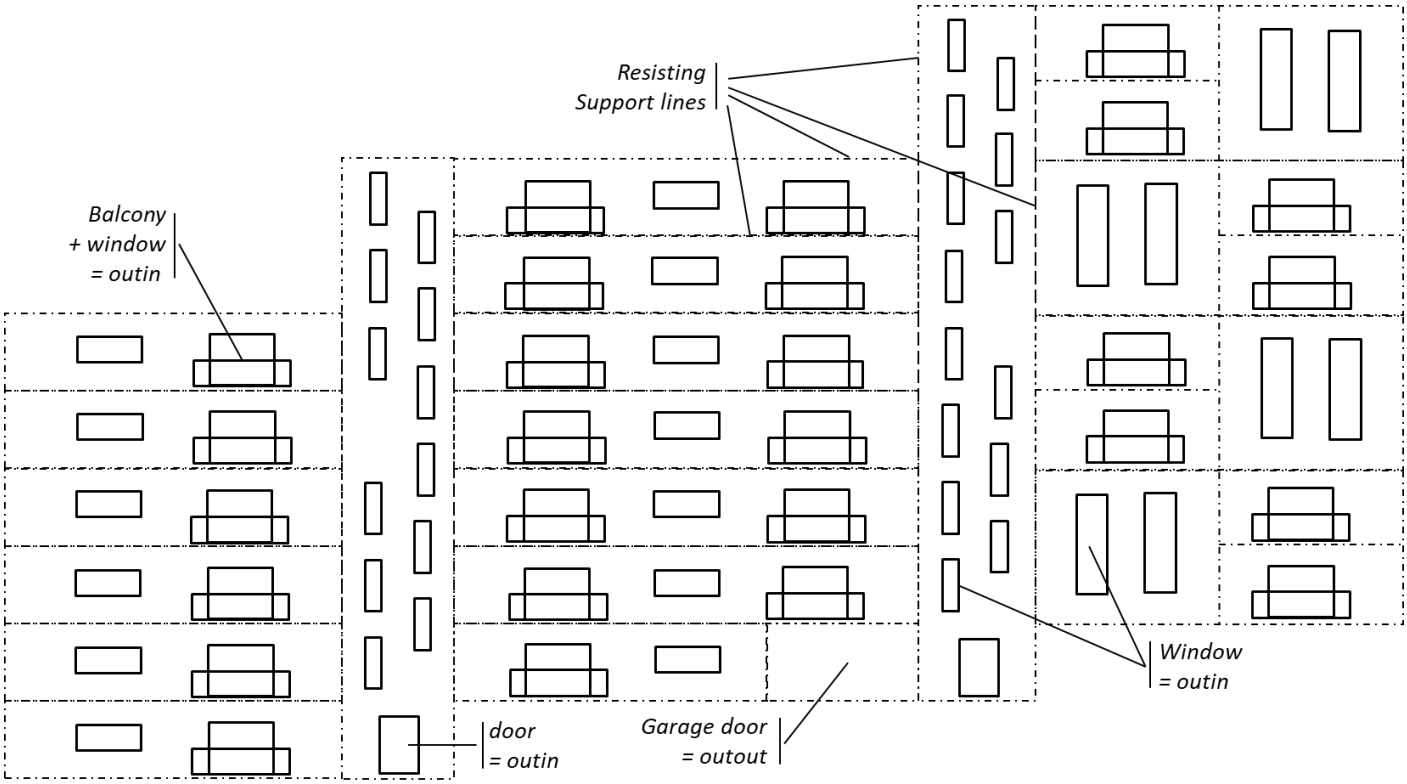

Figure 5. Exemple de façade prête pour le calepinage

inférieure, soit par ses coins inférieurs sur des *support line* verticale ou horizontale. Les panneaux peuvent avoir deux orientations, horizontale ou verticale, les panneaux horizontaux ont une largeur supérieure à leur hauteur, tandis que c'est le contraire pour les panneaux verticaux. Les fournisseurs de panneaux préfèrent travailler avec des panneaux horizontaux dans la mesure du possible.

Le problème est donc de trouver un ensemble de panneaux qui couvrent la façade. Comme de nombreuses solutions peuvent exister, différents critères peuvent être considérés. Les plus fréquents sont de minimiser le nombre de panneaux, maximiser la taille des panneaux ou minimiser la longueur des joints entre les panneaux. Ils poursuivent tous globalement le même objectif qui est de minimiser les pertes de chaleur.

4.2 Etat de l'art sur le calepinage

Compte tenu des éléments proposés, ce problème se classe suivant deux types de problèmes : *cutting and packing* [Kendal et al, 2010] et *product configuration* [Felfernig et al, 2014].

Les problèmes de *cutting and packing* considèrent de petits et de grands objets et tentent de placer les petits dans les grands. Les exemples classiques de tels problèmes sont : la production de barres et de plaques d'acier, la découpe de tissus, le chargement de véhicules, l'allocation de mémoire... Pour les problèmes à deux dimensions, comme notre problème de calepinage de panneaux, quatre caractéristiques ont un impact sur la complexité de la solution [Huang et Korf, 2012] : (i) la quantité de grands objets, 1 ou plus de 1, (ii) la quantité de petits objets, fournie en entrée ou à déterminer, (iii) la taille des petits objets, identiques ou non, et (iv) la taille des petits objets, fournie en entrée ou à déterminer.

Les approches de *product configuration* considèrent qu'un produit est un ensemble de composants prédéfinis (standard ou configurables) qui respectent diverses contraintes (provenant du marketing, de la conception, de la fabrication...). Les caractéristiques ayant un impact sur la complexité du problème sont : (i) la quantité de composants d'un produit : fournie en entrée ou à déterminer, (ii) le fait que les composants soient : standards (dimensions figées) ou configurables (dimensions à déterminer mais bornées) et (iii) la complexité des contraintes agissant sur l'ensemble de ces objets.

Pour ces deux approches, lorsque la quantité de petits objets et les dimensions pertinentes sont faibles et fournies en entrée du problème, il est le plus souvent possible de définir l'espace des solutions comme un arbre et de définir des algorithmes de recherche systématique. Lorsque la taille et/ou la complexité du problème sont modérées, il est alors envisageable de balayer l'espace complet des solutions et de déterminer des solutions optimales en combinant des algorithmes de recherche et de filtrage de contraintes. Lorsque nous quittons ce type de situations, il est beaucoup plus difficile de définir des algorithmes de recherche systématiques et la mise au point d'heuristiques devient la plupart du temps nécessaire.

Il apparaît que notre problème de calepinage de panneaux se situe vers le haut de l'échelle de taille et de complexité en raison : (i) des quantités, des dimensions et positions de tous les petits objets (panneaux) qui doivent être déterminées, (ii) de la diversité et la quantité des contraintes géométriques qui doivent être respectées. La caractéristique la plus délicate à gérer est le nombre de petits objets à déterminer (c'est-à-dire le nombre de panneaux). Cela implique effectivement que la quantité de variables du problème (ou taille du problème) n'est pas connue au moment du lancement de l'algorithme. Des auteurs comme [Barco et al, 2015] ont modélisé l'ensemble de ce problème comme un problème de satisfaction de contraintes et ont montré qu'il n'existait pas d'algorithme générique à base de contraintes capable de le traiter de manière systématique.

Etant donné que le travail présenté s'inscrit dans un projet à visée opérationnelle, il est nécessaire de fournir une solution robuste qui fonctionne avec toutes les caractéristiques précédentes. Nous avons donc conçu un algorithme basé sur une heuristique très proche du raisonnement humain. Cependant en complément, nous essayons actuellement de déterminer dans quelles mesures les solveurs basés sur les

contraintes peuvent traiter notre problème. La section suivante présente les idées principales de notre algorithme de calepinage automatique de panneaux.

4.3 Algorithme de calepinage

Après une petite introduction, nous décrivons la structure générale de l'algorithme, puis nous détaillons son fonctionnement pour les panneaux horizontaux. Nous terminons par quelques résultats expérimentaux et une discussion.

4.3.1 Paramètres clés de l'algorithme

Le fonctionnement de l'algorithme est modulé par deux paramètres clés :

- La stratégie d'orientation : HORIZONTALE ou VERTICALE. Ce paramètre définit l'orientation préférée des panneaux définis par l'utilisateur. Un panneau horizontal est fixé en bas et en haut sur des *support line* horizontales ou verticales. Un panneau vertical est fixé par ses 4 coins sur des *support line* horizontales ou verticales.
- La stratégie de direction : DROITE ou GAUCHE. Ce paramètre définit si l'ordre chronologique de placement et de dimensionnement des panneaux va de la gauche vers la droite ou le contraire.

Cependant, quel que soit le paramètre choisi, le placement dans son ensemble parcourt la façade du bas vers le haut, c'est-à-dire que les premiers panneaux seront positionnés en bas et les derniers en haut. Ceci s'apparente au processus d'assemblage sur site qui commence toujours par le bas et monte étage par étage.

Avant de lancer l'algorithme de calepinage, il est nécessaire de définir des points de départ initiaux. Ces points sont stockés dans une liste, qui sera remplie pendant le parcours de la façade, et supprimés une fois pris en compte lors du positionnement d'un panneau. Le prédicat de terminaison de l'algorithme est alors une liste vide. Initialement, la liste des points de départ est remplie avec chaque coin inférieur droit et gauche du périmètre de la façade.

4.3.2 Structure générale de l'algorithme

Le comportement général de l'algorithme suit une boucle qui s'exécute tant qu'il reste des points de départ dans la pile ordonnée de bas en haut (Algorithme 1). En cours d'exécution, il essaie de trouver un panneau suivant l'orientation et la direction souhaitées. Si la tentative échoue, un autre essai est effectué avec l'autre orientation possible. Par exemple, si l'algorithme est lancé avec les paramètres DROITE et HORIZONTAL, l'algorithme cherchera d'abord un panneau horizontal orienté vers la droite, et si ce n'est pas possible, un panneau vertical orienté vers la droite. Si un panneau est trouvé, il est ajouté à la solution qui sera renvoyée à la fin du déroulement de la boucle. De nouveaux points de départ possibles sont également ajoutés en fonction de chaque nouveau panneau dimensionné et positionné : pour la direction droite, les deux points ajoutés correspondent aux deux coins du panneau en haut à gauche et en bas à droite, tandis que pour l'autre direction, ce sont les deux coins en haut à droite et en bas à gauche.

La fonction addPanel() utilisée dans l'algorithme 1 est une simplification de présentation pour les fonctions addHorizontalPanel() et addVerticalPanel(). Ces deux fonctions sont très similaires et ne diffèrent légèrement que par leur technique de support et fixation. Les deux fonctions reçoivent un point de départ et ont pour objectif principal de trouver le meilleur second point définissant le plus grand panneau possible valide. Pour plus de clarté, seule la première fonction (pour les panneaux horizontaux) est détaillée dans le paragraphe suivant.

Algorithm 1 General layout design structure.

```
 1: Let Panel := x, y, width, height, out-ins
 2: Let p a Point variable x, y
 3: Let panel a panel variable
 4: direction ← RIGHT
 5: orientation ← HORIZONTAL
 6: startingPoints ← findStartingPoints(F, direction) // ordered stack
 7: Let solution := [Panel]
 8: solution = []
 9: while startingPoints is not empty do
10:     p ← selectNextPoint(startingPoints) // bottom to top, then direction
11:     panel = addPanel(p, F, cp, direction, orientation)
12:     if panel is null then
13:         panel = addPanel(p, F, cp, direction, other(orientation))
14:     end if
15:     if panel is null then throw Error (No solution found.)
16:     else
17:         solution.add(panel)
18:         addStartingPoints(startingPoints, panel)
19:     end if
20: end while
21: return solution
```

Lines 1 to 8 are variable definitions and input parameters:

1. A Panel data structure is defined (basically a rectangle with some out-ins)
2. A p variable of type Point
3. A panel variable of type Panel
4. Direction parameter set
5. Orientation parameter set
6. A set of points is calculated to start the algorithm, using one of the preparation algorithms
7. The type of the return value is set (an array of panels)
8. Return value set

4.3.3 Fonction d'ajout de panneau horizontal

La fonction de calepinage des panneaux horizontaux est divisée en deux étapes principales (voir Algorithme 2). Il construit d'abord une table de hachage des coordonnées possibles, où pour chaque y possible, les x possibles sont stockés (ligne 2). Ensuite, le plus grand panneau valide est trouvé en parcourant les coordonnées possibles ordonnées (lignes 25 à 35).

Pour la première étape, étant donné un point de départ p de coordonnées {xp, yp}, l'algorithme cherche d'abord une *support line* horizontale qui touche p. Ensuite, une collection géométrique de toutes les *support line* contiguës trouvées à partir de la première est rassemblée (findSupportLines() ligne 3). Il en résulte une *support line* horizontale avec toutes les coordonnées possibles X minimum et X maximum.

Nous stockons alors le maximum X (ou le minimum X si la direction est GAUCHE) dans la variable bottomMaxX (bottomMinX). Cela nous donne la plage [xp, bottomMaxX] ([bottomMinX, xp]) de toutes les coordonnées X possibles concernant le côté inférieur du panneau (lignes 5 et 16). Mais nous devons également tester les X possibles concernant le côté supérieur du panneau.

Ensuite, toutes les coordonnées Y possibles sont calculées, étant donné le point p initial. Pour chaque Y possible, on peut déduire le X maximum (X minimum) pour les *support line* inférieures et supérieures (en utilisant à nouveau findSupportLines() - lignes 9 et 19). Une fois ordonnées, les coordonnées possibles sont stockées.

La deuxième étape de l'algorithme est une double boucle qui essaie de construire et de retourner le plus grand panneau candidat valide. En ordonnant les coordonnées, nous nous assurons que le premier panneau valide trouvé sera le plus grand. La fonction valide évalue la validité des contraintes suivantes pour chaque panneau candidat :

- Validation des dimensions : les dimensions du panneau doivent respecter les valeurs minimales et maximales de largeur et de hauteur issues du principe constructif et des contraintes logistiques de transport,
- Validation des bordures : si la distance entre une bordure de panneau et un bord de façade ou un bord d'élément *outin* est supérieure à 0 mais inférieure à la largeur/hauteur minimale d'un panneau, le panneau n'est pas valide,
- Validation du confinement des éléments *outin* : les *outin* doivent être entièrement contenus dans un panneau,
- Validation du chevauchement avec d'autres panneaux : les panneaux ne doivent pas chevaucher les autres panneaux (ceux de la solution ou ceux placés manuellement initialement),
- Validation de la résistance : la résistance minimale des *support line* doit être conforme au poids du panneau (calculé à au prorata de sa surface).

Algorithm 2 Horizontal panel design algorithm.

```
1: function addHorizontalPanel(point, facade, cp, direction)→ Panel
2:         coordCandidates := Map{Y, [X]} = {} // possible x list for each y
3:     bottomSupportLines ← findSupportLines(point)
4:     if direction == RIGHT then
5:         bottomMaxX ← maxX(bottomSupportLines)
6:         possibleYs ← findPossibleYs(point)
7:         possibleYs ← possibleYs.reverse()
8:         for y in possibleYs do
9:                 maxX ← min(bottomMaxX, maxX(findSupportLines(point.x, y)))
10:            possibleXs ← findPossibleXs(point, maxX, y)
11:            possibleXs ← possibleXs.reverse()
12:            coordCandidates.put(y, possibleXs)
13:        end for
14:    else/* direction == LEFT */
15:        bottomMinX ← minX(bottomSupportLines)
16:        possibleYs ← findPossibleYs(point)
17:        possibleYs ← possibleYs.reverse()
18:        for y in possibleYs do
19:                minX ← max(bottomMinX, minX(findSupportLines(point.x, y)))
20:            possibleXs ← findPossibleXs(point, minX, y)
21:            coordCandidates.put(y, possibleXs)
22:        end for
23:    end if
24:    /* Browses possible coordinates for the 2nd point of panel */
25:    for x in coordCandidates.keys() do
26:        for y in coordCandidates[x] do
27:            panel ← createCandidate(point, x, y)
28:            if isValid(panel, facade, cp) then
29:                addResistingElements(panel)
30:                addFixingElements(panel)
31:                return panel
32:            end if
33:        end for
34:    end for
35:    return null
36: end function
```

4.3.4 Premiers résultats et discussion

Lorsque cet algorithme est utilisé avec l'exemple de la figure 5, une solution comprenant 34 panneaux est trouvée et représentée en figure 6 (les panneaux ont des coins arrondis pour les voir facilement). On peut voir que même si les panneaux horizontaux sont préférés, au moins 12 panneaux verticaux sont nécessaires pour les cages d'escaliers et les fenêtres des duplex à deux étages de la partie droite de la façade.

En termes d'optimalité, il faut souligner que l'orientation des panneaux a une influence majeure sur la solution et peut parfois diminuer le niveau d'optimalité d'une solution. Dans l'exemple de la figure 6, il est possible de voir que lorsque le panneau au bas de l'escalier de droite est conçu, l'orientation horizontale force un petit panneau horizontal (avec une porte au centre). Le plus souvent, les architectes préfèrent un seul grand panneau vertical qui remplacent les deux panneaux, comme le montre la figure 7.

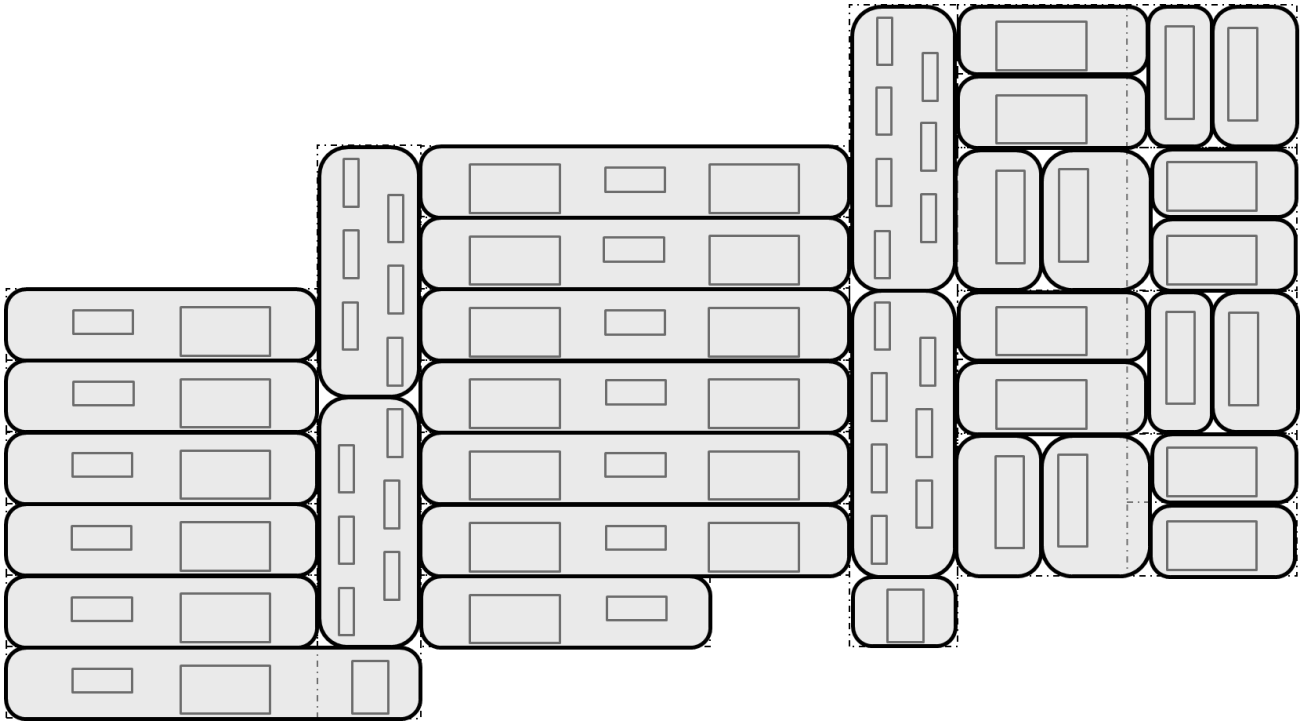

Figure 6. Solution de calepinage horizontale pour l'exemple de figure 5

Un algorithme plus optimal pourrait consister à calculer lors de chaque placement de panneau les deux alternatives (horizontale et verticale) et à conserver la meilleure. Cela pourrait être facilement implanté dans l'algorithme, mais cela conduirait à des solutions de calepinage mélangeant les deux orientations. Ces solutions peuvent être déroutantes car les panneaux ne semblent pas être disposés logiquement et sont délicats à gérer lors du montage sur site en raison de l'absence de répétabilité dans le mode assemblage. Cela pouvant conduire à des erreurs de montage de la solution.
C'est pour cette raison que nous avons conservé la prépondérance du paramètre d'orientation dans l'algorithme, mais avons cependant ajouté une dernière correction manuelle, à la fin du processus de conception après le calepinage automatique (figure 2). Au cours de cette dernière étape, des modifications manuelles peuvent être introduites afin de changer la taille de certains panneaux, de les regrouper ou d'aligner leurs bords. Cela permet d'améliorer l'optimalité de la solution et de prendre en compte les exigences spécifiques de l'architecte et/ou du propriétaire du bâtiment.

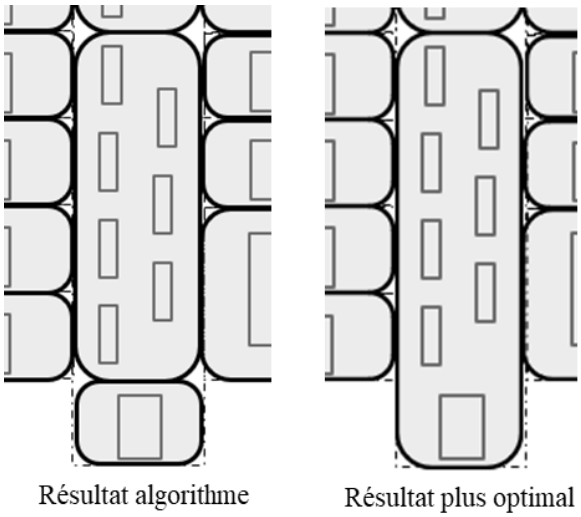

Figure 7. Un résultat plus optimal

Afin d'éviter le mélange hasardeux de panneaux d'orientations différentes, certains auteurs comme [Aldanondo et al, 2022] proposent de rechercher une décomposition de la façade en sous-façades entièrement horizontales ou verticales (c'est-à-dire n'acceptant que les panneaux horizontaux ou verticaux) avant de lancer l'algorithme de calepinage. L'intérêt principal de cette approche est de réduire la taille du problème de manière significative en le décomposant en sous-problèmes. Mais l'inconvénient est que certains cas ne peuvent pas être considérés. Par exemple, la partie droite de notre exemple ne peut pas être traitée avec des panneaux entièrement verticaux ou horizontaux. Nous pensons cependant que l'association de ces deux approches mériterait d'être étudiée.

La solution présentée en figure 6 a été obtenue avec une direction de calepinage ou de placement de panneau allant de la gauche vers la droite. Lorsque la direction opposée est utilisée (de la droite vers la gauche), la modification du calepinage ne concerne que la partie droite de la façade et est illustrée dans la figure 8. Il est à noter que c'est la présence d'outin (fenêtres et balcons) de grande dimensions horizontales et verticales qui nécessite de mélanger des panneaux avec les deux orientations.

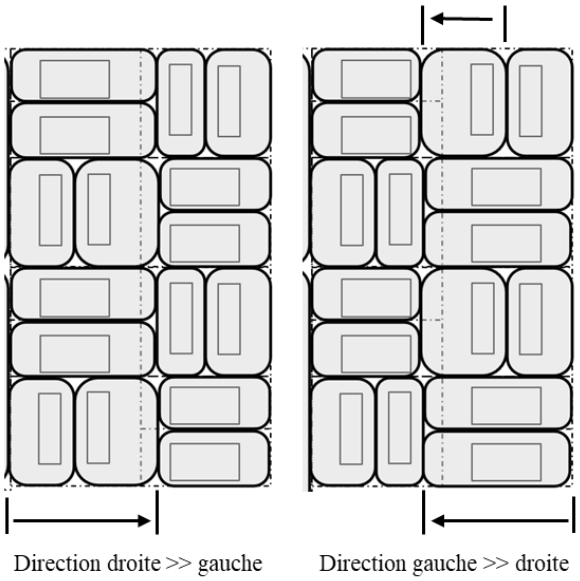

Figure 8. Comparaison différentes directions

5 CONCLUSION

Notre objectif était de présenter une approche globale pour aider le calepinage de panneaux paramétrables pour la rénovation de l'isolation des bâtiments. Nous avons proposé un processus en quatre étapes, trois modèles de données et un algorithme de calepinage qui permet de trouver de manière robuste des solutions avec une orientation préférentielle.

A notre connaissance, il n'existe pas dans les communautés proches du sujet, "cutting and packing" et "product configuration", d'approche exacte capable de modéliser et de résoudre de manière optimale le problème tel que défini. L'algorithme proposé est une heuristique gloutonne s'inspirant fortement du raisonnement humain dont le but est de concevoir des panneaux aussi grands que possible. Plus les panneaux sont grands, plus la quantité de panneaux et la longueur des joints diminuent. En conséquence directe, les performances énergétiques du bâtiment sont bien meilleures, ce qui répond à l'objectif final.

Au niveau opérationnel, le résultat est robuste, il a été utilisé sur différentes façades sans problème particulier. Le retour des partenaires du projet est satisfaisant, il est en effet possible de générer en moins d'une heure des solutions de disposition des panneaux qui nécessite manuellement au moins une journée de travail.

Actuellement, le problème étant formalisé comme un problème de satisfaction de contraintes, nous essayons d'utiliser le solveur à base de contraintes choco pour le reprogrammer afin d'avoir un programme plus déclaratif. Nous entendons par déclaratif une séparation des connaissances nécessaire au calepinage de l'algorithme lui-même. Actuellement cette connaissance est complétement imbriquée dans l'algorithme. Cela permettra, d'une part de modifier beaucoup plus facilement les contraintes relatives à la technologie des panneaux et au principe constructif des panneaux, actuellement le moindre changement nécessite le plus souvent de modifier l'algorithme. Par ailleurs, cela devrait probablement permettre d'améliorer l'optimalité des solutions.

6 REMERCIEMENTS

Les auteurs souhaitent remercier l'Agence Nationale de la Recherche française pour le financement du projet ISOBIM qui a permis la réalisation des travaux présentés.

7 REFERENCES

Aldanondo M., Vareilles E., Lesbegueries J., Christophe A. and Lorca X. (2022). Buildings energetic improvement: first elements about isolating panels layout design. In *Proceedings of the 14th IFAC Workshop on Intelligent Manufacturing Systems*, pages 487–492, Tel Aviv, Israel.

Aldanondo M., Barco-Santa A., Vareilles E. and Falcon M. (2014). Towards a BIM Approach for a High Performance Renovation of Apartment Buildings. In *Proceedings of the PLM 2014 conference*, Springer, pages 21–30 Yokohama, Japan, 2014.

Barco A., Vareilles E., Aldanondo M. and Gaborit P. (2014). A Recursive Algorithm for Building Renovation in Smart Cities. In *Lecture Notes in Computer Science*, LNAI Vol 8502 Springer pages 144–153, 2014.

Barco-Santa, A., Fages, J.G., Vareilles, E., Aldanondo, M., and Gaborit, P. (2015). Open packing for facade-layout synthesis under a general purpose solver. In *Proceedings of the 21rst CP conference* - Lecture Notes in Computer Science, vol. 9255, p. 508-523.

European Commission (2020). In focus: Energy efficiency in buildings. In https://ec.europa.eu/info/news/focus-energy-efficiency-buildings-2020-lut-17_en 2020.

Felfernig, A., Hotz, L., Bagley, C., and Tiihonen, J. (2014). Knowledge-Based Configuration: From Research to Business Cases. Morgan Kaufmann.

Hillebrand G., Arends G., Streblow R., Madlener R. and Müller D. (2014). Development and design of a retrofit matrix for office buildings. In *Energy and buildings*, Vol 70, pages 516-522, Elsevier, 2014.

Huang, E., and Korf, R.E. (2012). Optimal rectangle packing: An absolute placement approach. In *Journal of Artificial Intelligence Research*. Vol. 46, p. 47-87.

International Energy Agency. (2011). Prefabricated Systems for Low Energy Renovation of Residential Buildings. In IEA ECBCS Annex 50 or https://nachhaltigwirtschaften.at/resources/iea_pdf/iea_ecbcs_annex_50_anhang10b-moduledesign.pdf 2011.

Kendall, K., Daniels, K. and Burke, E. (2010). Cutting, packing, layout, and space allocation. In *Special issue Annals Operation Research*. n°179, p. 1-3.

Urbikain MK. (2020). Energy efficient solutions for retrofitting a residential multi-storey building with vacuum insulation panels and low-E windows in two European climates. In *Journal of Cleaner Production*, Vol 269, Elsevier.