

UNIVERSIDADE DE SÃO PAULO  
INSTITUTO DE CIÊNCIAS MATEMÁTICAS E DE COMPUTAÇÃO  
INSTITUTO DE ARQUITETURA E URBANISMO

CUSTOMIZAÇÃO DE MORADIAS: GERAÇÃO DE MODELOS BIM A  
PARTIR DE TÉCNICAS DE PROCESSAMENTO DE IMAGENS E  
VISÃO COMPUTACIONAL

Gustavo Henrique Brunelli

São Carlos  
2021

GUSTAVO HENRIQUE BRUNELLI

CUSTOMIZAÇÃO DE MORADIAS: GERAÇÃO DE MODELOS BIM A  
PARTIR DE TÉCNICAS DE PROCESSAMENTO DE IMAGENS E  
VISÃO COMPUTACIONAL

Relatório final apresentado à Universidade de São Paulo, como parte das exigências para a conclusão da Bolsa de Iniciação Científica, oferecida pelo Programa Unificado de Bolsas da Universidade de São Paulo, referente ao período de 01 de setembro de 2020 a 31 agosto de 2021.

Orientador: Prof. Dr. Márcio Minto Fabrício

São Carlos  
2021

## SUMÁRIO

<b>1</b>	<b>INTRODUÇÃO</b>	<b>5</b>
<b>1.1</b>	<b>OBJETIVOS</b>	<b>5</b>
<b>1.2</b>	<b>MÉTODO</b>	<b>6</b>
<b>2</b>	<b>DESENVOLVIMENTO DO PLUGIN</b>	<b>7</b>
<b>2.1</b>	<b>ESCOLHA DA PLATAFORMA</b>	<b>7</b>
<b>2.2</b>	<b>INTERFACE ENTRE SISTEMAS</b>	<b>7</b>
2.2.1	Paredes	7
2.2.2	Mobiliário	8
2.2.3	Janelas	9
2.2.4	Elementos Hospedados	9
2.2.5	Portas	10
<b>2.3</b>	<b>GERAÇÃO DE PISO, LAJE E TELHADO</b>	<b>11</b>
2.3.1	Ambientes	11
2.3.2	Piso e Laje	12
2.3.3	Telhado	12
<b>2.4</b>	<b>CLASSIFICAÇÃO DE AMBIENTES</b>	<b>15</b>
<b>2.5</b>	<b>COTAGEM DO PROJETO</b>	<b>15</b>
<b>3</b>	<b>INTERFACE GRÁFICA</b>	<b>17</b>
<b>3.1</b>	<b>FUNÇÕES</b>	<b>17</b>
<b>3.2</b>	<b>CONFIGURAÇÕES</b>	<b>18</b>
<b>4</b>	<b>ANÁLISE DOS RESULTADOS</b>	<b>20</b>

## LISTA DE FIGURAS

FIGURA 1 –Fluxo para gerar modelo digital . . . . .	5
FIGURA 2 –Elementos que acompanham o <i>template</i> . . . . .	8
FIGURA 3 –Possíveis orientações das portas . . . . .	10
FIGURA 4 –Telhado duas águas com beiral . . . . .	12
FIGURA 5 –Telhado Platibanda . . . . .	12
FIGURA 6 –Telhado várias água com beiral . . . . .	12
FIGURA 7 –Exemplo de cota gerada pelo plugin . . . . .	16
FIGURA 8 –Painel do Plugin . . . . .	17
FIGURA 9 –Janela para gerar projeto . . . . .	17
FIGURA 10 –Aba de elementos do menu de configurações . . . . .	18

## 1 INTRODUÇÃO

### 1.1 OBJETIVOS

Este projeto se insere em uma pesquisa mais ampla do grupo de pesquisa Arquitetura Inovação e Tecnologia do Instituto de Arquitetura e Urbanismo da Universidade de São Paulo - IAU USP, cuja finalidade é criar um artefato físico-digital para customização de moradias de interesse social em massa.

Desta forma, este projeto trata do desenvolvimento de um software capaz de gerar automaticamente um modelo BIM (*Building Information Modeling*), a partir de alguns parâmetros providos por um sistema de visão computacional, que por sua vez, recupera informações de uma maquete física (Figura 1).

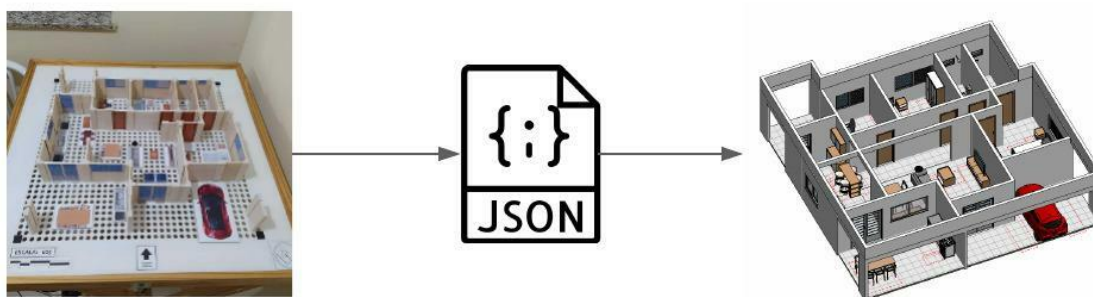


Figura 1 – Fluxo para gerar modelo digital

O software foi implementado por meio da API do Autodesk© Revit© 2021, como um plugin para a plataforma arquitetônica, contando com uma interface gráfica e parâmetros totalmente configuráveis pelo usuário.

O plugin consegue gerar todos os elementos definidos na maquete, como paredes, janelas, portas, mobília e algumas peças hidrossanitárias. E usando técnicas de geometria computacional, construir piso, laje e telhado a partir do perímetro do projeto. Além disso, também foram implementadas ferramentas para automatizar algumas tarefas, como a cotação do projeto, e um sistema de nomeação automática de ambientes com base em seus elementos.

## 1.2 MÉTODO

O autor escolheu por descrever o processo de desenvolvimento por funcionalidade, portanto, a ordem apresentada não necessariamente corresponde a ordem cronológica de implementação de cada componente. Os detalhes de implementação não serão tratados, no máximo, será apresentada as principais ideias e alguns algoritmos utilizados.

As ferramentas podem ser divididas em 4 grupos principais: criação dos elementos parametrizados que são fornecidos por meio de um arquivo JSON, que é resultado da leitura da maquete usando técnicas de visão computacional; geração de piso, laje e telhado a partir das paredes criadas; classificação de ambientes; cotagem de projeto.

## **2 DESENVOLVIMENTO DO PLUGIN**

### **2.1 ESCOLHA DA PLATAFORMA**

A escolha da linguagem foi guiada por quatro princípios: suporte, documentação, compatibilidade e praticidade. Visto que o Revit© utiliza o framework .NET (framework é um conjunto de código que oferece várias funcionalidades, o .NET, por exemplo, é proprietário da Microsoft©, e possui ferramentas para desenvolvimento no ambiente Windows©), é natural a escolha de uma linguagem com suporte nativo a essa ferramenta, as opções eram: VB, C++ e C#.

Como muitos dos tutoriais são publicados nesta linguagem, a documentação da API do Revit© ser em C#, por já ter tido contato com essa ferramenta, e por ser mais recente, a decisão final foi utilizar C#. Outra linguagem considerada foi o Python, mas pela necessidade de utilizar extensões não oficiais, o que gera uma dependência para atualizações e suporte, além de oferecer menos documentação, foi descartada apesar da possível agilidade de prototipagem e implementação.

### **2.2 INTERFACE ENTRE SISTEMAS**

A interface entre o plugin e o sistema de visão computacional é feita por meio de um arquivo no formato JSON, que contém os parâmetros necessários para a construção de cada elemento no Revit©. Os elementos estão divididos nos seguintes grupos: paredes, mobiliário, janelas, elementos hospedados e portas.

#### **2.2.1 Paredes**

O Revit© fornece diversos métodos para a inserção das paredes no modelo BIM, o escolhido foi pelo ponto inicial e final de cada parede. Assim, no arquivo JSON, cada parede possui duas coordenadas correspondentes às suas extremidades. O algoritmo para a criação de paredes no documento BIM consiste em criar uma reta parametrizada que funciona como guia para a parede, e fornecer informações como altura, tipo de material e nível.

### 2.2.2 Mobiliário

Para instanciar um objeto de mobília, que no Revit® é uma instância de uma família, é necessário informar o ponto no plano cartesiano no documento onde ele deve ser inserido, a sua rotação e o tipo do objeto. No JSON são passados dois pontos, já que na maquete física os elementos de mobília são definidos por duas letras, a sua rotação em graus, e o seu tipo, que é um código de 2 ou 3 caracteres.

Para transformar essas informações em parâmetro válidos para a API do Revit®, é calculado o ponto médio entre os pontos fornecidos, e a rotação é convertida para radianos. Para encontrar qual objeto o código representa, foi implemento um banco de dados relacional local, que armazena os códigos dos elementos e o nome do seu tipo no *template*.

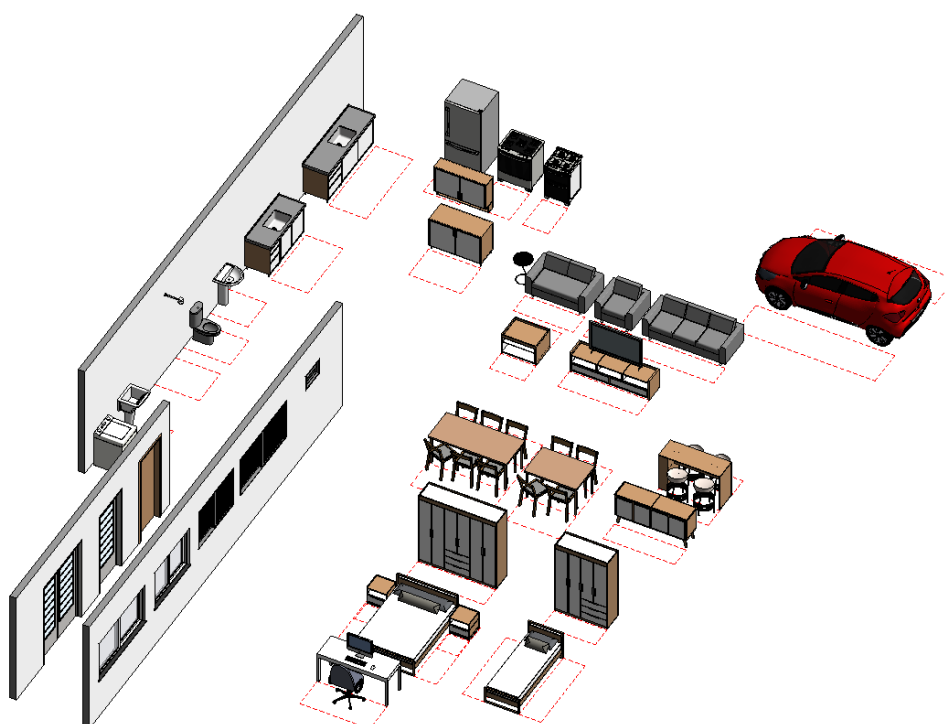


Figura 2 – Elementos que acompanham o *template*

Um problema que surgiu usando a abordagem por pontos, é que muitas vezes as letras na maquete física não correspondiam exatamente ao centro do objeto, e quando essas eram instanciados do Revit®, saíam deslocados. A solução para esse problema foi relacionar a cada entrada na tabela de elementos no banco de dados, um vetor configurável para fazer a correção de deslocamento, assim, é feito uma soma entre esse vetor com o ponto em que o JSON fornece. A base de coordenadas do vetor



é modificada de acordo com a rotação de cada elemento, para que o deslocamento seja feito referente a rotação que o objeto será ser inserido, não à base de coordenadas do projeto como um todo.

### 2.2.3 Janelas

A inserção de janelas é feita por meio de um elemento hospedeiro, que nesse caso é uma parede. Para instanciar uma janela, é necessário uma referência da parede que ela deve ser hospedada, um parâmetro  $t$  da reta guia da parede, e o tipo da janela. Para encontrar a parede hospedeira, foi usado um algoritmo de busca linear, que itera para todas as paredes definidas no projeto, e retorna a parede mais próxima no ponto  $P$ , informado pelo JSON.

A própria API do Revit© fornece métodos que auxiliam no cálculo do parâmetro  $t$  da reta, assim não foi necessário o desenvolvimento de nenhum algoritmo para calcular esse parâmetro. Para obter o tipo da janela, foi usado o mesmo princípio da mobília. Cada janela possui uma linha da tabela de elementos, com seu código, nome de tipo, e vetor de deslocamento.

### 2.2.4 Elementos Hospedados

Os elementos hospedados são objetos que também são fixados em uma parede, mas ficam sobre a face na parede, não interna a ela, como por exemplo: vaso sanitário, pia e chuveiro. O arquivo JSON fornece apenas um único ponto para essa classe de elementos e o tipo. A inserção do elemento funciona de forma similar às janelas: por meio de uma busca pela parede hospedeira.

Porém, o método que a API do Revit© fornece para inserir elementos hospedados não acopla o elemento do lado correto da parede, ele sempre será inserida da direção do vetor normal da linha da parede. Assim, foi necessário implementar um algoritmo que corrige a direção do elemento. Sendo  $\vec{n}$  o vetor normal da parede, e  $P$  o ponto a ser inserido dado pelo JSON, calculamos o ponto  $P_w$  que é ponto da parede mais próximo ao ponto  $P$ . Considere  $\gamma_p$  (2.1) a reta formada pelo ponto  $P$  e o vetor  $\vec{n}$ , e  $\gamma_w$  a reta guia da parede.

$$\gamma_p(\lambda) = P + \lambda \vec{n} \quad (2.1)$$

Então obtemos o ponto da parede no ponto  $\vec{P}'$  em que  $\gamma_p = \gamma_w$  para algum

valor de  $\lambda$ . Então é calculada a suposta direção  $\vec{d}$  do objeto hospedado (2.2), e por fim, a direção que o objeto foi inserido é comparada com a suposta direção  $\vec{d}$  que o objeto deveria ter, se forem diferentes, invertemos o objeto hospedado através de métodos oferecidos pela API do Revit®. Com o elemento já inserido, aplicamos os deslocamentos de correção relacionado ao hospedado.

$$\vec{d} = \frac{\vec{P} - \vec{P}'}{\|\vec{P} - \vec{P}'\|} \quad (2.2)$$

### 2.2.5 Portas

As portas são parametrizadas pelo ponto que deve ser posta no projeto, o lado de abertura, e a mão de abertura. Depois que a porta já foi inserida pelo método já descrito para janelas e hospedados, é recuperado o vetor  $\vec{l}$  de abertura fornecido pela API, e comparado com o lado de abertura fornecido pelo JSON, se forem diferentes, invertemos a porta.

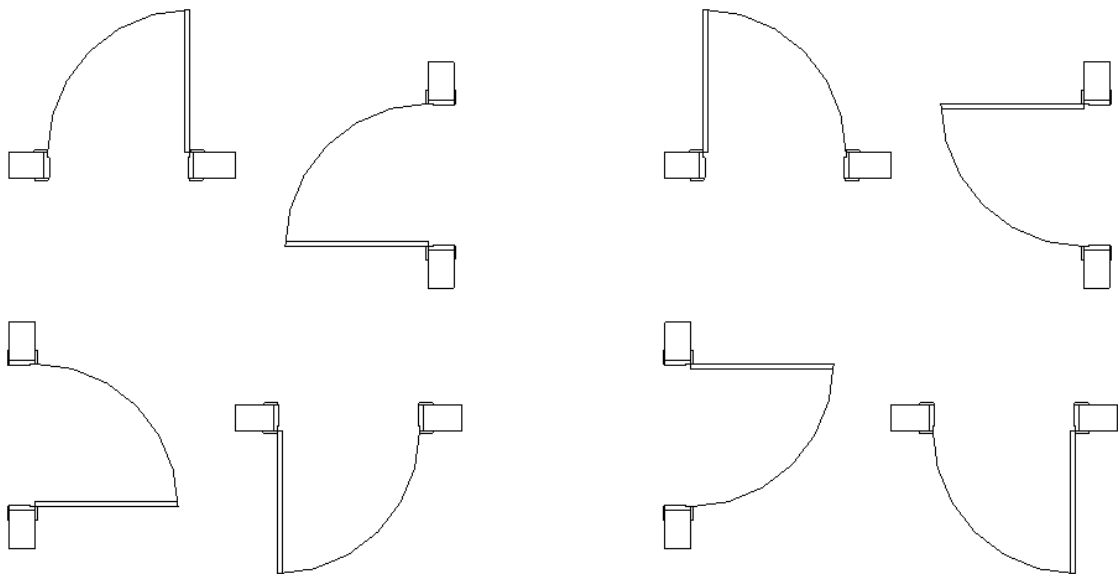


Figura 3 – Possíveis orientações das portas

Para definirmos a mão de abertura, é feito o produto vetorial  $\vec{u} = \vec{m} \times \vec{l}$ , sendo  $\vec{m}$  o vetor que representa a mão da porta. Caso a componente Z do vetor  $\vec{u}$  seja positiva, dizemos que a mão da porta é direita, caso contrário dizemos que é esquerda. Então fazemos a inversão caso a mão indicada pelo JSON for da mão em que a porta foi inserida.

## 2.3 GERAÇÃO DE PISO, LAJE E TELHADO

### 2.3.1 Ambientes

Para entender como são gerados os pisos, a laje e o telhado da casa, é necessário entender o funcionamento de ambientes no Revit®. Um ambiente é uma região definida por um polígono qualquer, normalmente definidos por paredes de um cômodo, mas também podem ser delimitados por divisores de ambientes. Um divisor de ambiente é um segmento reta, que funciona como uma parede invisível, o que é extremamente útil, já que o método de recuperar perímetros, ou como o Revit® os chama: circuitos, é feito pelas linhas das paredes e divisores de ambientes.

Contudo, os ambientes e divisores não são automaticamente postos no projeto apenas por construir paredes que formam circuitos. Assim, foi necessário desenvolver um algoritmo que faz esse trabalho de colocar os divisores e ambientes em locais adequados.

Para colocar os divisores, é feita uma iteração sobre cada parede do projeto, e todas as paredes que possuírem uma das extremidades não conectada com outra parede são marcadas como candidatas para gerar um separador de ambientes. Para cada uma dessas paredes  $w_i$ , chamamos de  $P_d$  os pontos desconectados (a parede pode ter as duas extremidades desconectadas, funcionando como uma coluna), e  $P_c$  o ponto conectado.

Para cada um dos pontos desconectados, calculamos o vetor  $\vec{u} = \vec{P}_d - \vec{P}_c$ , e geramos uma reta parametrizada  $\gamma_w(\lambda) = P_d + \lambda\vec{u}$ . Então iteramos para as todas as outras paredes do projeto, procurando a parede mais próxima cuja reta guia intercepte a reta  $\gamma_w$ , e que não esteja conectada com a parede  $w$ , caso não encontre nenhuma parede na direção de  $\vec{u}$ , procuramos as paredes que interceptam a reta  $\delta(\lambda) = P_d + \lambda\vec{v}$ , sendo  $\vec{v}$  um vetor perpendicular a  $\vec{u}$ . Se encontrarmos uma parede, definimos um divisor de ambientes entre  $\vec{P}_d$  e o ponto de intersecção da reta  $\gamma_w$  ou  $\delta_w$  com a reta guia da parede encontrada.

Com os divisores de ambientes já colocados, podemos definir os ambientes da casa. A API fornece um método que retorna todos os planos de circuito, que são os circuitos fechados da casa, cada plano de circuito contém os circuitos daquele plano. Assim, podemos iterar sobre cada plano de circuito, e definir um ambiente sobre ele.

### 2.3.2 Piso e Laje

O piso da casa é inserido em todos os ambientes separadamente, para dar a possibilidade do usuário alterar o seu material manualmente, caso deseje, de um ambiente individual. A laje, porém, é gerada em um único bloco, seguindo o perímetro da casa.

A solução usada para calcular o perímetro externo da casa requer que o terreno seja fechado por muros. Isso se torna necessário pois para encontrar o perímetro da casa, é feita uma iteração sobre todos os ambientes do documento, e é verificado o número de circuitos de cada ambiente. Caso o ambiente tenha 2 circuitos: o do muro e o da construção, o chamamos de Externo, e o perímetro da casa é o circuito desse ambiente com menor área.

### 2.3.3 Telhado

O plugin oferece a opção de gerar um telhado que segue a geometria da casa, em três opções diferentes de estilo: telhado com duas águas e várias águas com beiral, e platibanda. O plugin usa um algoritmo com quatro estágios de pipeline: normalização de perímetro, decomposição do perímetro em polígonos convexos, aplicação de beiral e construção da geometria do telhado.

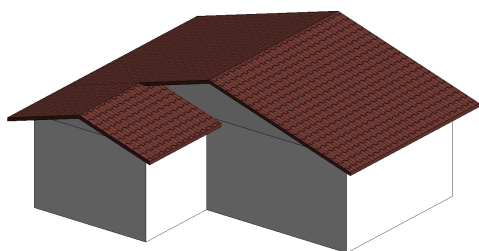


Figura 4 – Telhado duas águas com beiral

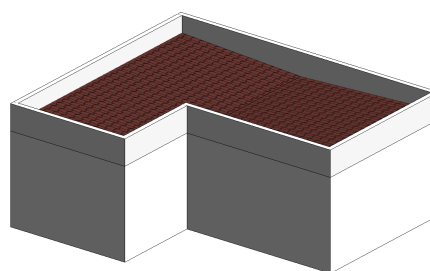


Figura 5 – Telhado Platibanda

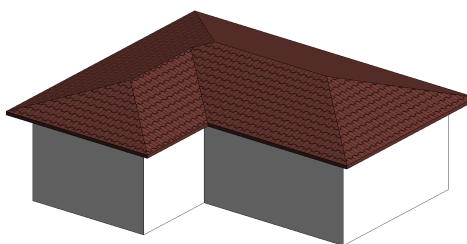


Figura 6 – Telhado várias água com beiral

Como o JSON oferece as definições da parede como estão na maquete, muitas vezes a parede é gerada em vários pedaços, o que pode ser um problema para os algoritmos seguintes, por isso, antes de tudo é feita uma normalização do perímetro do telhado. Isso é feito eliminando vértices desnecessários que ligam duas arestas paralelas.

Na segunda etapa, o perímetro é dividido em polígonos mais simples, por meio de um algoritmo de decomposição. O algoritmo funciona do seguinte modo: primeiramente é identificado os vértices que formam um ângulo interno maior do que  $180^\circ$ , que serão chamados de "entalhe". O objetivo é eliminar os entalhes por traçar retas partindo dele para outro vértice ou aresta, dividindo assim o polígono em seus componentes convexos.

Para cada um dos entalhes  $E$ , definimos uma reta  $\gamma(\lambda) = E + \lambda\vec{n}$ , em que  $\vec{n}$  é a direção do corte definido pelo usuário. Então escolhemos o ponto  $P$  pertencente a reta  $\gamma$ , que faça parte do polígono da iteração atual, mais próximo de  $E$ . Traçamos o segmento  $\overline{PE}$ , e separamos o polígono inicial por esse segmento. Executamos novamente o algoritmo para cada um dos polígonos gerados, até não haver nenhum entalhe.

O algoritmo de decompor polígonos retorna todas as linhas de corte que foram feitas, para que na etapa de expansão, essas retas não sejam alteradas, visto que elas representam intersecções de geometria do telhado, e se sofrerem alterações, vão estragar a forma do telhado.

A seguir, aplicamos o seguinte algoritmo para expandir o polígono: seja  $V_j$  um vértice qualquer do polígono,  $V_i$  o vértice antecessor, e  $V_k$  o vértice sucessor, no sentido positivo. Calculamos os vetores  $\vec{v}_1 = \vec{V}_j - \vec{V}_i$  e  $\vec{v}_2 = \vec{V}_k - \vec{V}_j$ , e caso algum desses vetores correspondam a uma linha de corte, ele não é alterado nos passos seguintes.

Considerado um valor real  $b$  como sendo o tamanho em metros do beiral, geramos uma reta paralela ao vetor  $\vec{v}_1$  e  $\vec{v}_2$ , com uma distância  $b$  de  $V_i$ . Para isso, primeiramente calculamos

$$\vec{u}_1 = \frac{\vec{v}_1}{\|\vec{v}_1\|}b \text{ e } \vec{u}_2 = \frac{\vec{v}_2}{\|\vec{v}_2\|}b.$$

Seja  $\vec{u}_1 = x_1\vec{i} + y_1\vec{j}$  e  $\vec{u}_2 = x_2\vec{i} + y_2\vec{j}$ , então definimos

$$\vec{n}_1 = -y_1\vec{i} + x_1\vec{j} \text{ e } \vec{n}_2 = -y_2\vec{i} + x_2\vec{j},$$

O ponto inicial de cada reta será

$$P_1 = V_i + \vec{n}_1 \text{ e } P_2 = V_k + \vec{n}_2,$$

E as equações das retas serão

$$\gamma_1(\lambda) = P_1 + \lambda\vec{n}_1 \text{ e } \gamma_2(\lambda) = P_2 + \lambda\vec{n}_2.$$

Então calculamos o ponto onde  $\gamma_1 = \gamma_2$ , esse ponto será o vértice com o espaçamento aplicado que substituirá  $V_j$ . Repetimos esse procedimento para todos os vértices de cada polígono gerados anteriormente. Com o formato dos componentes do telhado já definidos, a próxima etapa é construir a geometria do telhado.

Para o telhado de empena, o usuário escolhe para quais direções a queda deve ser feita, através de uma interface gráfica. Assim, a queda é aplicada para todos os lados cuja reta guia é perpendicular a direção dada. As paredes de empena são construídas nas mesmas retas, porém no local que estavam antes de sofrerem o deslocamento pelo algoritmo de expansão.

Para o telhado de várias águas, a queda é aplicada para todas as retas da extremidade do telhado. Para construir esse tipo de telhado, não é necessário que o polígono que o representa sofra o processo de decomposição em componentes convexos.

Para o telhado platibanda, calculamos o centroide  $C$  do seu polígono, e o dividimos em dois polígonos pela reta  $\gamma(\lambda) = C + \lambda\vec{n}$ , sendo  $\vec{n}$  um vetor perpendicular à direção de queda definida pelo usuário na interface gráfica. Depois são construídas as paredes platibanda com o mesmo perfil do telhado.

## 2.4 CLASSIFICAÇÃO DE AMBIENTES

O plugin conta com uma funcionalidade de classificação automática de ambientes. Essa classificação nomeia os ambientes da casa com base no mobiliário que há dentro dele. O funcionamento desse sistema se dá por meio de um algoritmo de pontuação, assim, cada elemento previsto na maquete tem uma pontuação dada por um número inteiro, para cada ambiente.

Por exemplo, o chuveiro pode somar cinco pontos para o ambiente "Banheiro", já uma pia de cozinha pode somar cinco pontos para o ambiente "Cozinha", e três pontos para "Varanda de Lazer". Também é possível determinar números negativos para a pontuação, o que é útil para definir ambientes muitos parecidos, que se diferenciam por apenas um ou dois elementos.

Esse sistema é implementado por meio de um banco de dados local, que faz uma relação do tipo muitos para muitos entre a tabela de elementos, já citada, e uma tabela de ambientes, que também é totalmente configurável pelo usuário através de uma interface gráfica. A pontuação entre a relação é atribuída em uma tabela de associação de elementos e ambientes.

O algoritmo calcula a pontuação daquele ambiente para cada uma das possibilidades de nomeação definida pelo usuário, e escolhe aquela cuja pontuação é a maior. O ambiente externo, que é definido na etapa de cálculo de perímetro, não é alterado.

Através do menu de configurações do plugin, é possível customizar totalmente a classificação de ambientes, por inserir novas relações, alterar relações e pontuações existentes, assim como adicionar novos elementos e novos ambientes.

## 2.5 COTAGEM DO PROJETO

Também foi implementada uma função para cotar automaticamente o projeto gerado. A cotação é feita em referência às paredes externas da casa, de acordo com a definição da maquete. Por usar a medida das paredes da maquete, é comum que paredes contínuas apareçam cortadas na cotagem. Para contornar esse problema, duas cotas são feitas, uma medindo as paredes cortadas, e outra, mais externa, medindo o segmento todo. A segunda cota é feita por aplicar uma normalização no perímetro da casa.

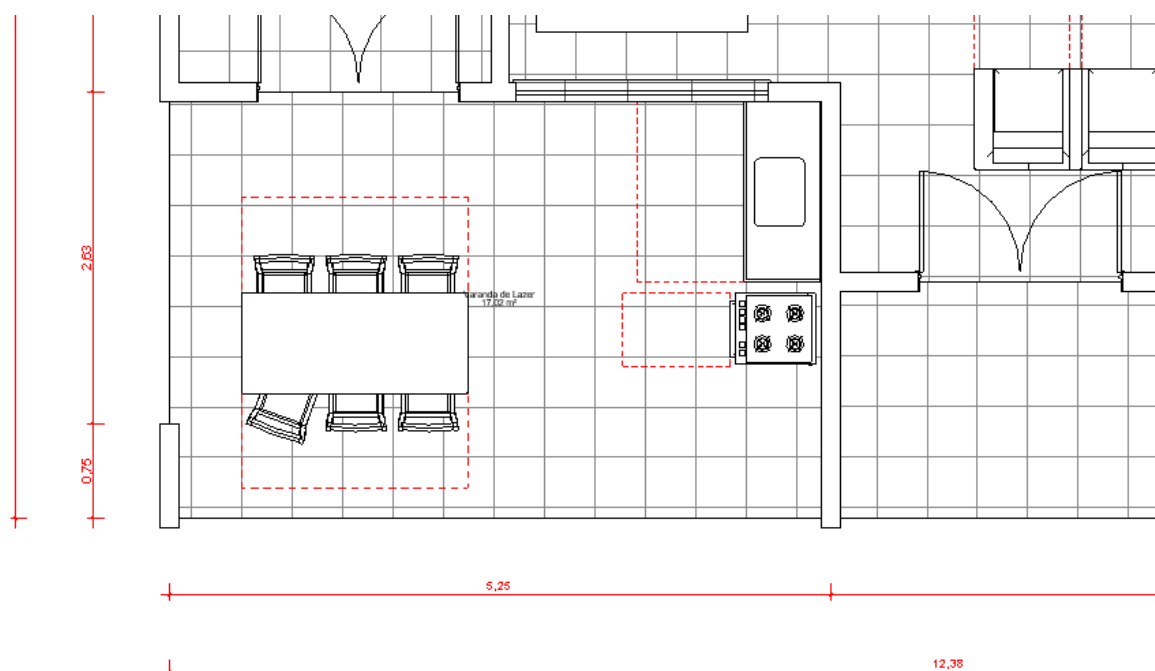


Figura 7 – Exemplo de cota gerada pelo plugin



### 3 INTERFACE GRÁFICA

#### 3.1 FUNÇÕES

O plugin é integrado no painel de ferramentas do Revit®, fornecendo uma interface simples e de fácil uso. O painel é dividido em 3 sessões: geração completa, funções e geral.

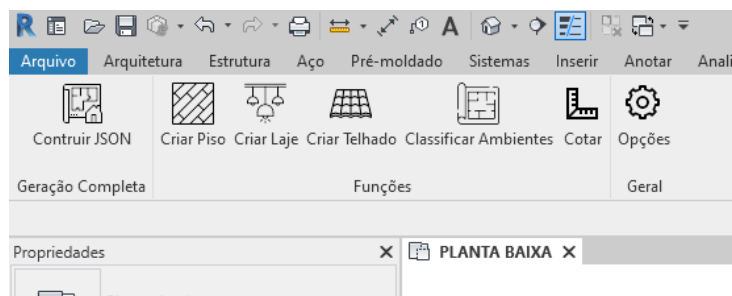


Figura 8 – Painel do Plugin

Na geração completa, há um botão para construir a moradia por completo, desde os elementos descritos no JSON, até o piso, laje, telhado e ambientes, e quando pressionado, dispara um evento que abre uma janela para selecionar o arquivo JSON, e escolher o tipo e o lado da queda do telhado.

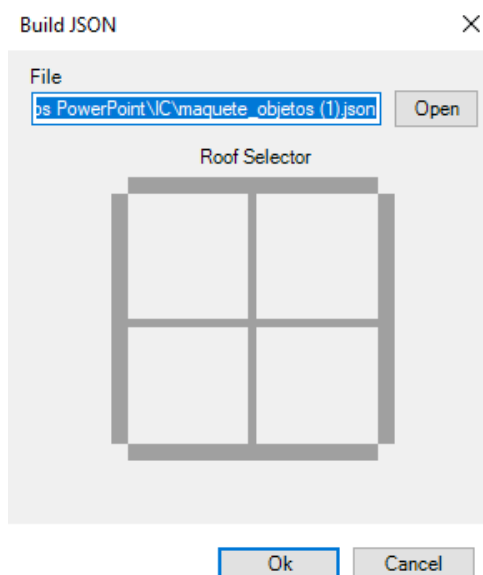


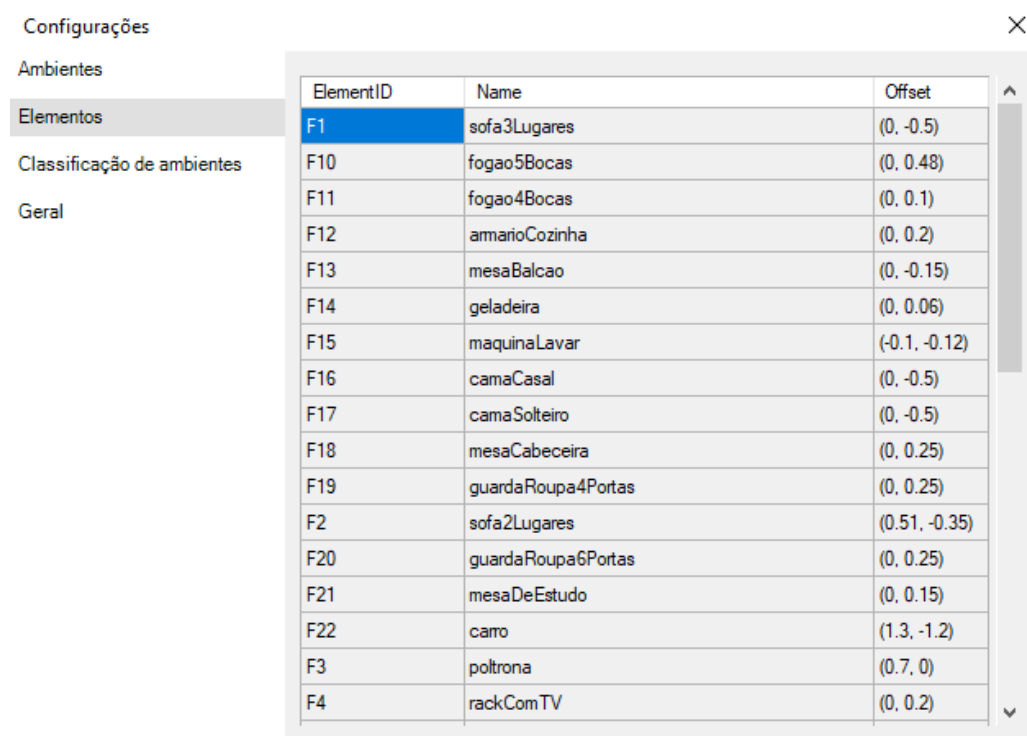
Figura 9 – Janela para gerar projeto

Já na sessão de funções, são fornecidas as funcionalidades separadamente, caso o usuário queira, por exemplo, executar uma em específico depois de alterar algum

detalhe do projeto. Essa opção é especialmente útil para telhados, já que o projeto é enviado para o Revit© em dois “lotes”, o primeiro é toda a construção, e a segunda é o telhado, isso permite que o usuário desfaz a última ação após criar o projeto, somente o telhado seja removido, permitindo que ele teste outras gerações.

### 3.2 CONFIGURAÇÕES

Na sessão geral, há um botão de opções, que abre um menu que permite configurar boa parte dos parâmetros do plugin. Na aba de ambientes, é possível adicionar ou remover ambientes. Para adicionar, basta clicar na última linha em branco, escrever o nome do novo ambiente, e pressionar *Enter*, para remover, é necessário clicar duas vezes na linha no nome do ambiente e apagar o texto.



ElementID	Name	Offset
F1	sofa3Lugares	(0, -0.5)
F10	fogao5Bocas	(0, 0.48)
F11	fogao4Bocas	(0, 0.1)
F12	amarioCozinha	(0, 0.2)
F13	mesaBalcao	(0, -0.15)
F14	geladeira	(0, 0.06)
F15	maquinaLavar	(-0.1, -0.12)
F16	camaCasal	(0, -0.5)
F17	camaSolteiro	(0, -0.5)
F18	mesaCabeceira	(0, 0.25)
F19	guardaRoupa4Portas	(0, 0.25)
F2	sofa2Lugares	(0.51, -0.35)
F20	guardaRoupa6Portas	(0, 0.25)
F21	mesaDeEstudo	(0, 0.15)
F22	carro	(1.3, -1.2)
F3	poltrona	(0.7, 0)
F4	rackComTV	(0, 0.2)

Figura 10 – Aba de elementos do menu de configurações

Na aba de elementos, é possível visualizar os códigos dos elementos da coluna ID, o seu nome, que é o tipo de determinado de família definida no Revit©, e o vetor de deslocamento, que será aplicado ao inserir o elemento na geração. Todas essas informações são configuráveis de forma similar aos ambientes. Para remover, basta deletar o nome e o ID de um elemento.

Na aba de classificação de ambientes, é possível alterar, remover e adicionar novas relações entre ambientes e elementos. Ao clicar sobre um elemento ou ambiente, será aberto uma janela para selecionar um elemento ou ambiente previamente definido nas suas respectivas abas.

Por fim, na aba de configurações gerais, é possível modificar parâmetros globais do plugin. O nível base deve ser o nível em que as paredes serão construídas, o nível do telhado será o nível em que a laje, o telhado serão inseridos, e também define a altura das paredes. O tipo de piso, laje e parede devem estar especificados no documento Revit®.

## 4 ANÁLISE DOS RESULTADOS

A construção dos elementos parametrizados fornecida pelo processo de visão computacional é bem robusto, e consegue lidar com os mais diversos tipos de projetos. O sistema de deslocamento de objetos foi extremamente satisfatório para corrigir problemas de posicionamento das marcações dos elementos físicos, gerado por limitações da maquete física.

O algoritmo de correção de orientação de portas e objetos hospedados funcionam bem para os elementos fornecidos junto ao template, porém pode falhar caso o usuário queira inserir novos elementos que não sigam os mesmo padrões dos elementos do template.

A geração de de piso, laje e telhado também funciona para a maioria dos projetos, porém pode apresentar ocasionais falhas. Em específico, a geração de telhado de duas águas com beiral costuma apresentar alguma dificuldade para construir telhado em projetos muito exóticos, devido a sua complexidade.

A classificação de ambiente apresentou resultados surpreendentemente bons, acertando com uma altíssima acurácia o nomes dos ambientes em relação ao idealizado pelo usuário. Como a interface de configuração possibilitando customizar por completo a nomeação, e o sistema de inserção de separadores de ambientes, é uma das ferramentas mais robustas desse plugin.

A ferramenta de cotagem do projeto, apesar de funcional e não apresentar falhas, ainda não está de acordo com as convenções usadas na arquitetura, e pode ser melhorada para remover medidas redundantes, e simplificação de cotagem de segmentos de paredes.

O resultado final foi um plugin bastante robusto, porém ainda apresenta algumas falhas e necessita bastante cuidado do usuário ao manipular a maquete para projetar casas que são geradas como ele havia imaginado. E as principais limitações do plugin atualmente são a necessidade do projeto no Revit© ser cercado por muros, e a impossibilidade de criar paredes na diagonal.

Assim, ao projetar uma habitação, recomenda-se que o usuário se atente no alinhamento das paredes, privilegiando projetos com um perfil mais simples. E se necessário fazer projetos mais complexos, dar preferência a geração de telhado do tipo platibanda ou várias águas, que possuem menos ocorrências de erros.