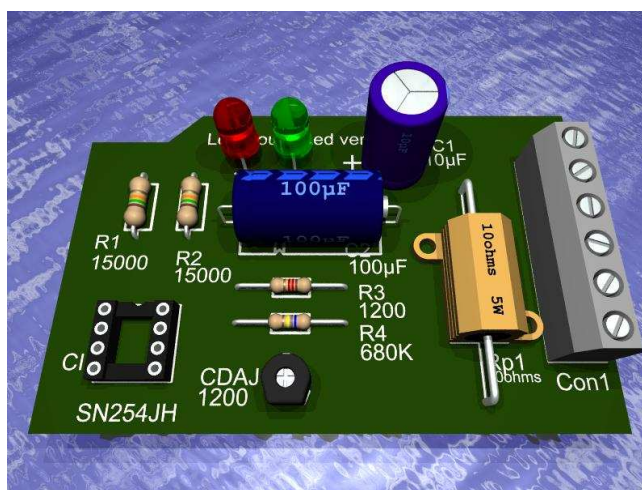


Logiciel Wintypon : Wintypon & la 3D

(Génération d'une vue 3D du circuit)

Création d'un modèle 3D



**Attention: Cette documentation est la suite de la documentation:
Aide CAO 04 - Wintypon 3D - Présentation & mode d'emploi**

Il est impératif de la lire auparavant !

*Cette documentation présente en détail
la création d'un modèle 3D d'un transformateur type EE20*

Transformateurs moulés Type : EE20 - 0,5 VA



A partir de
5.15€

Version du logiciel Wintypon: 7.0 et plus

Version de cette documentation : 1.35

Date : 15 mai 2010

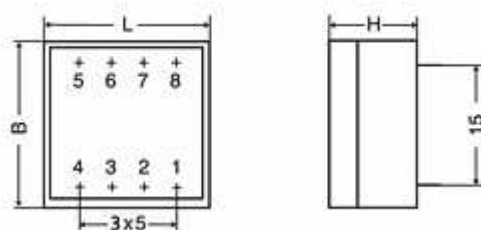
Auteur de cette documentation: Mr EYNARD Pascal - Auteur Wintypon (Correction: Alain)

Logiciels Wintypon & Visu3D : www.typonrelais.com

Editeur : Circuit imprimé jurassien

A – Création de l'empreinte du transformateur

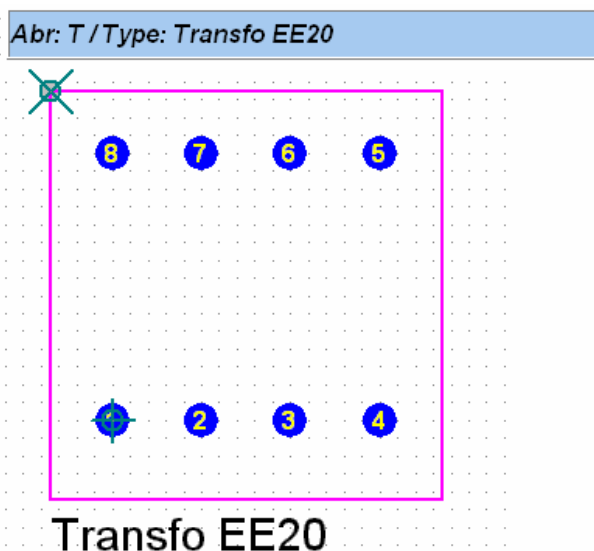
Il faut impérativement posséder la documentation constructeur, pour avoir les bonnes dimensions.



Les dimensions du transformateur

Pour ce modèle ; B = 23 mm, L = 22 mm et H = 19 mm

Dans le logiciel Empreinte, l'empreinte est créée, sur une grille au pas de 1mm (et non pas 2.54mm, puisque ce transfo est au pas de 5 mm). Le composant étant dessiné en vue de dessus, les broches sont numérotées en conséquence (Attention aux vues des fabricants, de dessus, de dessous...!).



B – La création du modèle 3D

Comme tous les modèles 3D, il y a 3 étapes à respecter :

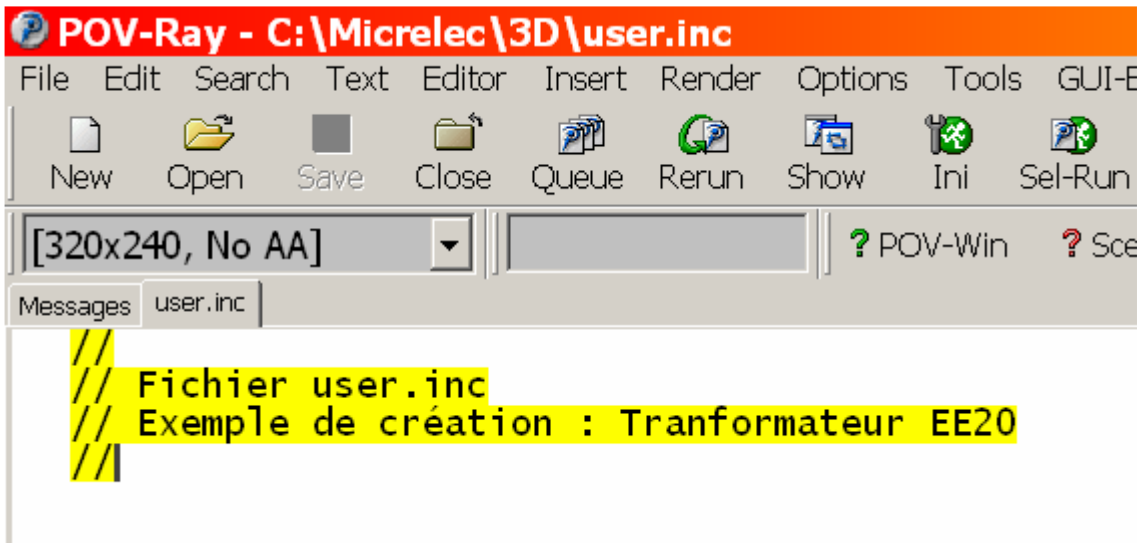
- 1 - Définir une macro d'appel, dans le fichier USER.INC
- 2 - Associer ce modèle avec l'empreinte de Wintypon.
- 3 - Terminer le modèle.

Etape 1 – Définir une macro d'appel, dans USER.INC

Il est fortement déconseillé de modifier un des fichiers INC fourni (ic.inc, diode.inc...), afin de ne pas perdre son travail en cas de mise à jour.

Il faut donc utiliser USER.INC. Il est là pour cela. Il est fourni vide.

Ouvrir donc avec Povray le fichier USER.INC (dossier ..\program files\elec-cao\3D)



Il faut choisir un nom pour la macro d'appel.

Le nom " Transfo_EE20 " sera choisi.

Note : Il existe des recommandations pour le choix des noms de macros. Car normalement, après avoir créé un modèle, vous l'envoyez à moi-même (auteur de wintypen - www.typonrelais.com) et/ou à Matthias Weisser (auteur de Eagle 3D - www.matwei.de). Et pour intégrer votre modèle à la librairie déjà existante, il est bien de respecter certaines conventions.

Ces recommandations sont en annexe 1 à la fin de cette documentation.

Rappel : Tous les modèles 3D sont disponibles et communs aux utilisateurs de Wintypen et Eagle™ 3D. La création d'un modèle est donc appréciée. Il s'agit d'un projet ouvert à tous ☺.

Dans Povray, cela donne :

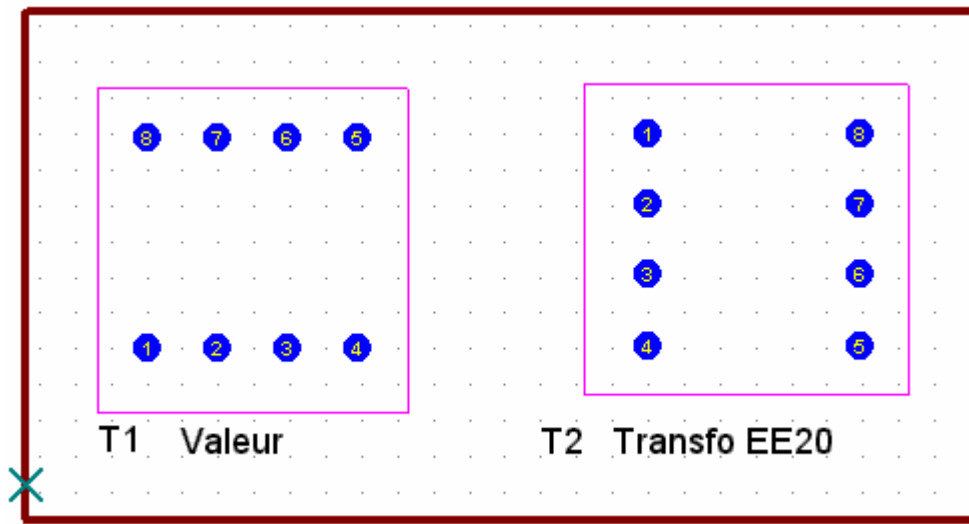
```
//  
// Fichier user.inc  
// Exemple de création : Transformateur EE20  
//  
  
#macro TRANSFO_EE20(Valeur)  
  
.. le modèle sera décrit ici..  
  
#end
```

Cette macro a un unique paramètre d'entrée (entre parenthèses) : Valeur (= la valeur du composant).
Le " #end " indique la fin de la macro.

Une macro ne peut être vide (Povray génère une erreur dans ce cas).

Nous allons donc définir une boîte de 10 mm de côté, couleur argent. C'est juste pour mettre quelque chose...

```
//  
// Fichier user.inc  
// Exemple de création : Transformateur EE20  
//
```

Le typon avec 2 transformateurs EE20 posés



La vue 3D du typon, avec le modèle non terminé

Maintenant il reste à dessiner vraiment le modèle.

Etape 3 – Terminer le modèle

A ce niveau là, seul le langage Povray entre en compte. Si besoin, les liens suivants vous aideront à maîtriser les bases de ce langage.

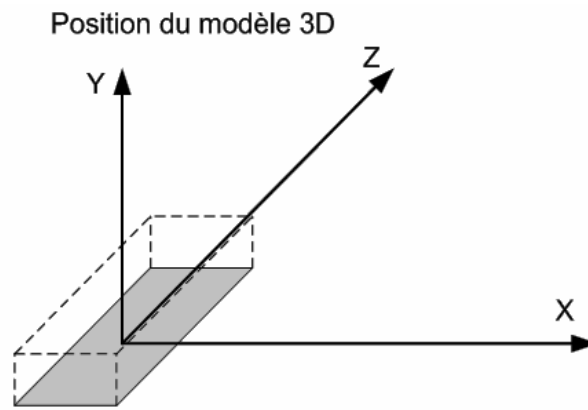
www.povray.org
<http://pov.monde.free.fr/>
<http://perso.orange.fr/sebastien.bancquart>
<http://www.lightning-generator.org/>
<http://povwiki.ovh.org/>
<http://www.3dvh.com>

Site principal (en anglais)
 Site francophone - Aide de povray traduite - liens - Docs..
 Didacticiel complet en français
 Site francophone très riche - Docs, liens..
 Un wiki sur Povray, très très riche.
 Portail français sur la 3D en général

Le transformateur fait 22 * 23 * 15 mm. De plus, il faut que le modèle soit centré autour du point [0,0].

Soit en coordonnées Povray : X = 22, Y = 15, Z = 23.

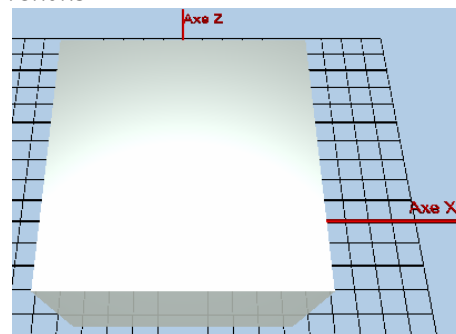
Attention: Sous Povray, Y est la hauteur, Z la profondeur. Le modèle doit s'élever en hauteur à partir de Y=0. La hauteur Y=0 est le dessus du circuit imprimé.



Position du modèle 3D dans le repère Povray

Donc la box va devenir : `box <-11,0,-11.5> <11,15,11.5>`, nous obtenons :

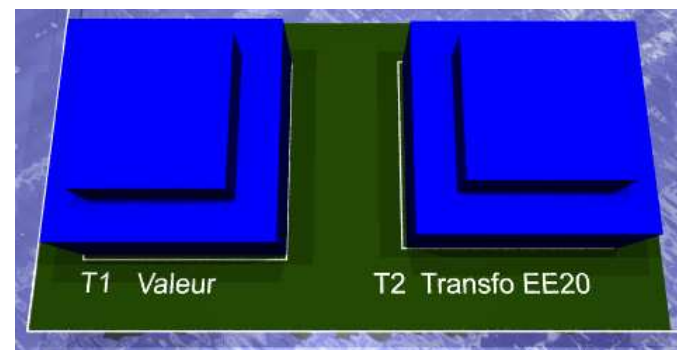
```
#macro TRANSFO_EE20(Valeur)
union {
box <-11,0,-11.5> <11,15,11.5> }
texture{col_silver}
}
#end
```



Le composant est bien centré, et posé sur le plan [Z,X].

En fait, il y a un décrochement, à environ 4 mm sous le sommet. Nous allons donc réduire la hauteur de cette box, et en ajouter une plus petite dessus. Et changer la couleur en bleu. Le résultat est vu sur le typon de test.

```
#macro TRANSFO_EE20(Valeur)
union {
box <-11,0,-11.5> <11,10,11.5> }
box <-7,10,-7.5> <7,15,7.5> }
pigment {Blue}
}
#end
```

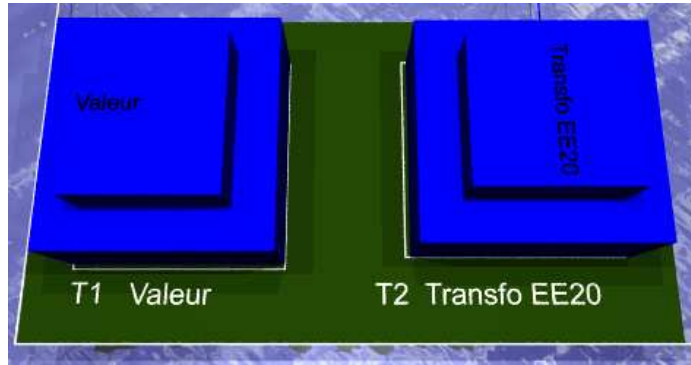


Le modèle commence à ressembler à un transformateur ☺.

Pour une 1^{ère} approche, ce réalisme suffit. Nous améliorerons cela plus tard.

Il reste à placer la valeur qui sera écrite sur le transformateur.

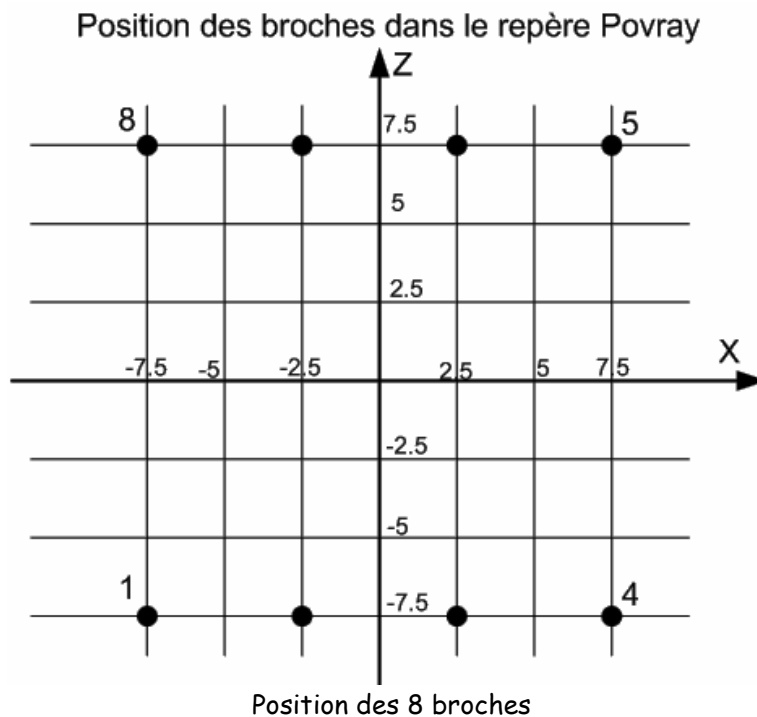
```
#macro TRANSFO_EE20(Valeur)
union {
box <-11,0,-11.5> <11,10,11.5> }
box <-7,10,-7.5> <7,15,7.5> }
box<<-0.1,0,0><0.1,0.03,22> translate<39.64,0,8.86>}
text{ttf "Arial" Valeur 0.1,0 scale<2,2,1> rotate<90,0,0> translate<-6,15.1,0> pigment {Black} }
pigment {Blue}
}
#end
```



La valeur inscrite sur le dessus.

Pour terminer, il reste les 8 broches à placer.

Positionnons ces broches dans le repère Povray (vue de dessus donc). Les broches sont bien centrées autour du point [0,0], comme le composant.



Les broches seront des cylindres verticaux, de rayon 0.25mm, et de longueur 9 mm.

Broche 1 : $X = -7.5$ $Y = Z = -7.5$. La hauteur ira de $Y=-8$ à $Y=1$ (la broche va donc dépasser de 8 mm sous le composant, et " rentrer " de 1 mm à l'intérieur - mais cela ne se voit pas).

La texture sera col_silver (argent), ce qui donnera :

```
cylinder{<-7.5,-8,-7.5><-7.5,1,-7.5>0.25 texture {col_silver} }
```

Idem pour les 7 autres broches:

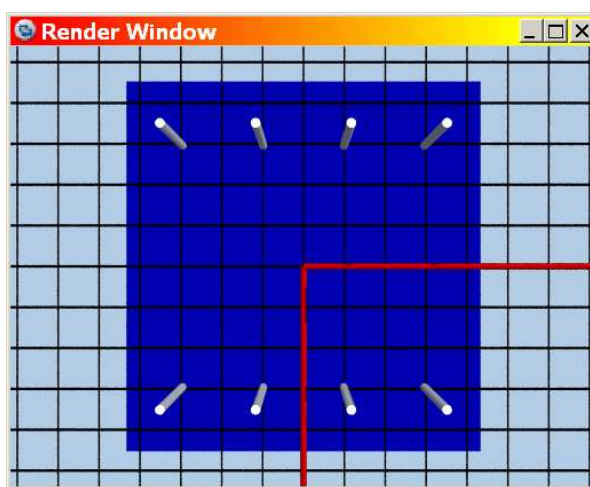
```
cylinder{<-7.5,-8,-7.5><-7.5,1,-7.5>0.25 texture {col_silver} }
cylinder{<-2.5,-8,-7.5><-2.5,1,-7.5>0.25 texture {col_silver} }
cylinder{<2.5,-8,-7.5><2.5,1,-7.5>0.25 texture {col_silver} }
cylinder{<7.5,-8,-7.5><7.5,1,-7.5>0.25 texture {col_silver} }
```

```
cylinder{<-7.5,-8,7.5><-7.5,1,7.5>0.25 texture {col_silver} }
cylinder{<-2.5,-8,7.5><-2.5,1,7.5>0.25 texture {col_silver} }
cylinder{<2.5,-8,7.5><2.5,1,7.5>0.25 texture {col_silver} }
cylinder{<7.5,-8,7.5><7.5,1,7.5>0.25 texture {col_silver} }
```


Ce qui donne au final :

```
#macro TRANSFO_EE20(Valeur)
union {
box {<-11,0,-11.5> <11,10,11.5> }
box {<-7,10,-7.5> <7,15,7.5> }
box{<-0.1,0,0><0.1,0.03,22> translate<39.64,0,8.86>}
text{ttf "Arial" Valeur 0.1,0 scale<2,2,1> rotate<90,0,0> translate<-6,15,1,0> pigment {Black} }
cylinder{<-7.5,-8,-7.5><-7.5,1,-7.5>0.25 texture {col_silver} }
cylinder{<-2.5,-8,-7.5><-2.5,1,-7.5>0.25 texture {col_silver} }
cylinder{<2.5,-8,-7.5><2.5,1,-7.5>0.25 texture {col_silver} }
cylinder{<7.5,-8,-7.5><7.5,1,-7.5>0.25 texture {col_silver} }
cylinder{<-7.5,-8,7.5><-7.5,1,7.5>0.25 texture {col_silver} }
cylinder{<-2.5,-8,7.5><-2.5,1,7.5>0.25 texture {col_silver} }
cylinder{<2.5,-8,7.5><2.5,1,7.5>0.25 texture {col_silver} }
cylinder{<7.5,-8,7.5><7.5,1,7.5>0.25 texture {col_silver} }
pigment {Blue}
}
#end
```

Vérifions cela en générant une vue de dessous, zoomée (Z+) avec Visu3D :



Vue de dessous

Tout va bien. Les broches sont bien positionnées. ☺

La création du modèle 3D de ce transformateur peut être considérée comme terminée. Le niveau de réalisme est correct. Mais il est possible de faire mieux...

Nous allons donc poursuivre en découvrant certains points avancés:

- L'option " pin_short " : Limitation de la longueur des broches.
- L'amélioration du réalisme: Création de bords arrondis.
- Création de 2 macros pour la conception des broches.

3a - Option " pin_short "

Il existe plusieurs options pour modifier l'aspect de la vue 3D du circuit. La liste complète est en annexe 2.

L'option pin_short permet de limiter la longueur des broches à une valeur pin_length (pin_length étant aussi une variable...).

En l'utilisant, toutes les broches de tous les composants du circuit ont alors la même longueur. Cette longueur sera alors égale à pin_length + pcb_height (Longueur des broches si pin_short = on + hauteur du circuit).

Les options et variables sont définies au début des fichiers POV générés par Visu3D et Wintypou:

```
#declare pin_length = 2.5;           ( Longueur des broches sous le circuit )
#declare pin_short = on;             ( Limiter la longueur sous le circuit à pin_length )
```

Pour changer la hauteur des broches, au lieu de mettre $Y = -8$, il suffit de mettre $Y = -(pin_length+pcb_height)$.

Il faut également employer un test: `if pin_short = on then....elseend`.

Cela donne au final :

```
#macro TRANSFO_EE20(Valeur)
union {
box {<-11,0,-11.5> <11,10,11.5> }
box {<-7,10,-7.5> <7,15,7.5> }
box{<-0.1,0,0><0.1,0.03,22> translate<39.64,0,8.86>}
text{tff "Arial" Valeur 0.1,0 scale<2,2,1> rotate<90,0,0> translate<-6,15,1,0> pigment {Black} }

#if(pin_short=off)
cylinder{<-7.5,-8,-7.5><-7.5,1,-7.5>0.25 texture {col_silver} }
cylinder{<-2.5,-8,-7.5><-2.5,1,-7.5>0.25 texture {col_silver} }
cylinder{<2.5,-8,-7.5><2.5,1,-7.5>0.25 texture {col_silver} }
cylinder{<7.5,-8,-7.5><7.5,1,-7.5>0.25 texture {col_silver} }
cylinder{<-7.5,-8,7.5><-7.5,1,7.5>0.25 texture {col_silver} }
cylinder{<-2.5,-8,7.5><-2.5,1,7.5>0.25 texture {col_silver} }
cylinder{<2.5,-8,7.5><2.5,1,7.5>0.25 texture {col_silver} }
cylinder{<7.5,-8,7.5><7.5,1,7.5>0.25 texture {col_silver} }

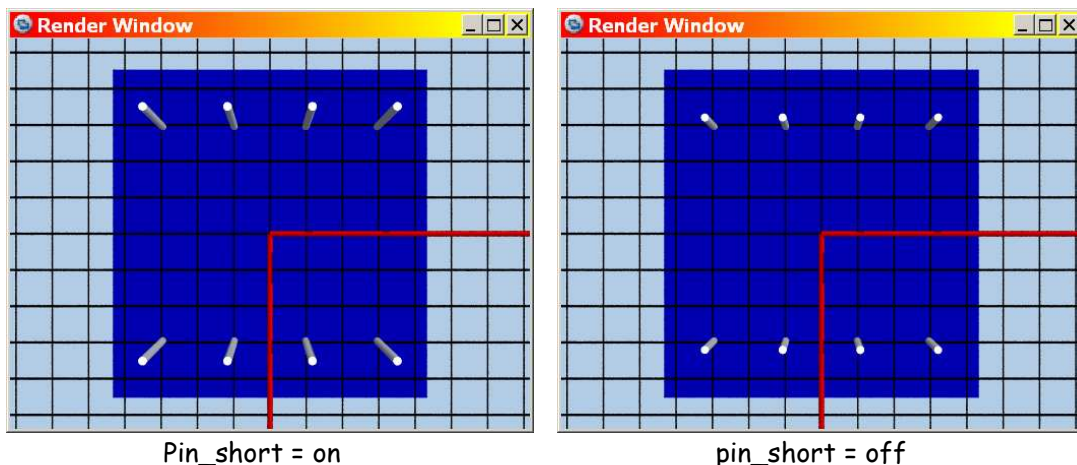
#else

cylinder{<-7.5,-(pin_length+pcb_height),-7.5><-7.5,1,-7.5>0.25 texture {col_silver} }
cylinder{<-2.5,-(pin_length+pcb_height),-7.5><-2.5,1,-7.5>0.25 texture {col_silver} }
cylinder{<2.5,-(pin_length+pcb_height),-7.5><2.5,1,-7.5>0.25 texture {col_silver} }
cylinder{<7.5,-(pin_length+pcb_height),-7.5><7.5,1,-7.5>0.25 texture {col_silver} }
cylinder{<-7.5,-(pin_length+pcb_height),7.5><-7.5,1,7.5>0.25 texture {col_silver} }
cylinder{<-2.5,-(pin_length+pcb_height),7.5><-2.5,1,7.5>0.25 texture {col_silver} }
cylinder{<2.5,-(pin_length+pcb_height),7.5><2.5,1,7.5>0.25 texture {col_silver} }
cylinder{<7.5,-(pin_length+pcb_height),7.5><7.5,1,7.5>0.25 texture {col_silver} }

#end

pigment {Blue}
}
#end
```

Ce qui en vue de dessous donne :



Ce code est fonctionnel, mais il comporte 16 lignes presque semblables... ce n'est pas optimal. De plus, tous les composants ont des broches, nous verrons donc plus loin comment utiliser des macros pour optimiser cela.

3b - Amélioration du réalisme: Création de bords arrondis

La forme de base est la boîte (box), pour faire des arrondis, il faut complètement redéfinir cette forme. Les arrondis sont réalisés avec des sphères & des cylindres.

On voit que ce transformateur comporte en fait 2 boîtes arrondies (sur le dessus seulement), l'une au dessus de l'autre. Donc plutôt que de définir 2 boîtes arrondies semblables, mais de dimensions différentes, il est plus malin de faire une macro pour définir une boîte arrondie de dimensions quelconques, puis d'utiliser 2 fois cette macro. Cette macro se nommera: WT3D_BOX_ROUND_SUP_GRND.

Remarque : Dès qu'une forme complexe est présente plusieurs fois, il est plus malin de faire une macro pour cet objet. Puis l'utiliser ensuite. C'est plus simple, plus clair, et surtout, la macro peut servir à d'autres modèles, plus tard. Il est inutile de réinventer la roue à chaque modèle...

La structure du fichier USER.INC va donc devenir:

```
#macro WT3D_BOX_ROUND_SUP_GRND (tailleX,tailleY,tailleZ,rayon) → La macro
...
    ( Définition de la macro qui trace une boîte arrondie de taille fixée )
...
#end

#macro TRANSFO_EE20(Valeur)
union {
...
object {WT3D_BOX_ROUND_SUP_GRND (22,15,23,2) } → Utilisation de la macro
...
#end
```

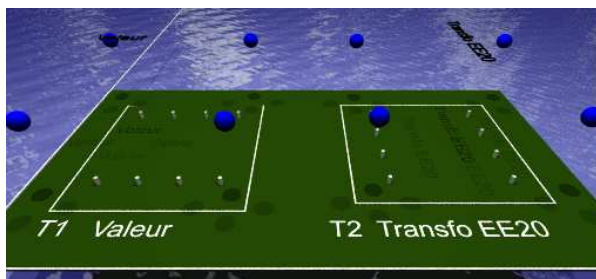
Cette macro va avoir 4 paramètres d'entrées:

- tailleX = Dimension X
- tailleY = Dimension Y
- tailleZ = Dimension Z
- rayon = rayon de l'arrondi

Réalisation de la macro boîte arrondie : WT3D BOX ROUND SUP GRND:

Nous commençons par placer 4 sphères dans les angles hauts:

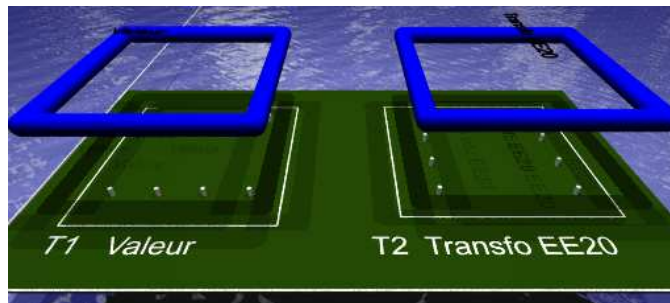
```
sphere {<-tailleX/2 + rayon, tailleY - rayon,-tailleZ/2 + rayon> rayon }
sphere {<-tailleX/2 + rayon, tailleY - rayon, tailleZ/2 - rayon> rayon }
sphere {< tailleX/2 - rayon, tailleY - rayon,-tailleZ/2 + rayon> rayon }
sphere {< tailleX/2 - rayon, tailleY - rayon, tailleZ/2 - rayon> rayon }
```



Ajout des 4 sphères

puis 4 cylindres horizontaux:

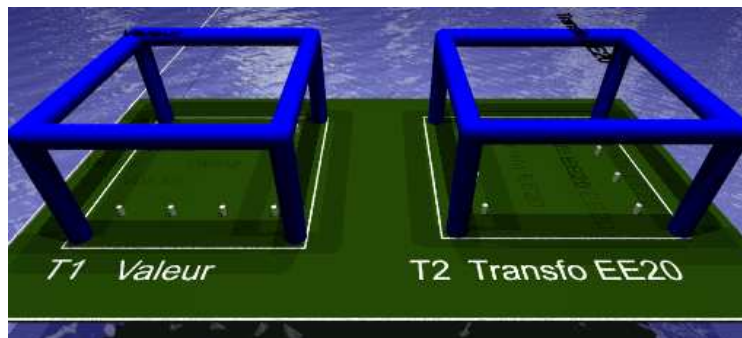
```
cylinder {<-tailleX/2 + rayon, tailleY - rayon,-tailleZ/2 + rayon><-tailleX/2 + rayon, tailleY/1 - rayon, tailleZ/2 - rayon> rayon }
cylinder {< tailleX/2 - rayon, tailleY - rayon, tailleZ/2 - rayon>< tailleX/2 - rayon, tailleY/1 - rayon,-tailleZ/2 + rayon> rayon }
cylinder {<-tailleX/2 + rayon, tailleY - rayon,-tailleZ/2 + rayon>< tailleX/2 - rayon, tailleY/1 - rayon,-tailleZ/2 + rayon> rayon }
cylinder {<-tailleX/2 + rayon, tailleY - rayon, tailleZ/2 - rayon>< tailleX/2 - rayon, tailleY/1 - rayon, tailleZ/2 - rayon> rayon }
```



Ajout des 4 cylindres horizontaux

puis 4 cylindres verticaux:

```
cylinder {<-tailleX/2 + rayon,0 ,-tailleZ/2 + rayon><-tailleX/2 + rayon, tailleY - rayon,-tailleZ/2 + rayon> rayon }
cylinder {<-tailleX/2 + rayon,0 , tailleZ/2 - rayon><-tailleX/2 + rayon, tailleY - rayon, tailleZ/2 - rayon> rayon }
cylinder {< tailleX/2 - rayon,0 ,-tailleZ/2 + rayon>< tailleX/2 - rayon, tailleY - rayon,-tailleZ/2 + rayon> rayon }
cylinder {< tailleX/2 - rayon,0 , tailleZ/2 - rayon>< tailleX/2 - rayon, tailleY - rayon, tailleZ/2 - rayon> rayon }
```

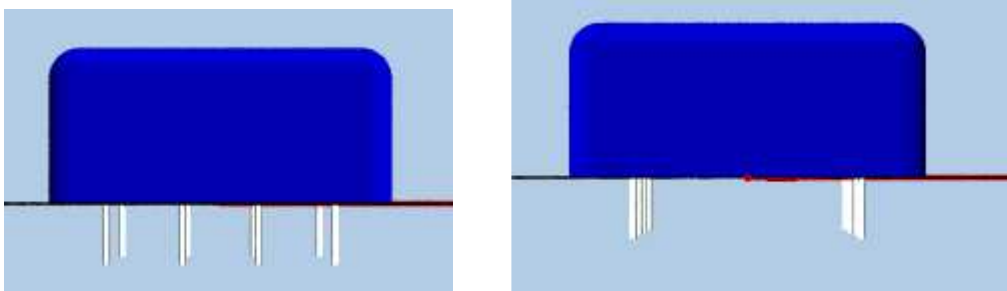


Ajout des 4 cylindres verticaux

et enfin nous fermons la structure avec 3 box:

```
box {<-tailleX/2 + rayon,0,-tailleZ/2><tailleX/2-rayon,tailleY-rayon,tailleZ/2> }
box {<-tailleX/2,0,-tailleZ/2+rayon><tailleX/2,tailleY-rayon,tailleZ/2-rayon> }
box {<-tailleX/2 + rayon,0,-tailleZ/2+rayon><tailleX/2-rayon,tailleY,tailleZ/2-rayon> }
```

Le résultat est visualisé dans Visu3D, pour bien vérifier toutes les faces:



Vérification des faces avec Visu3D

La macro WT3D_BOX_ROUND_SUP_GRND semble donc correcte, et permet de dessiner une boîte arrondie sur le dessus de taille quelconque.

Le code complet de cette macro est en annexe 3.

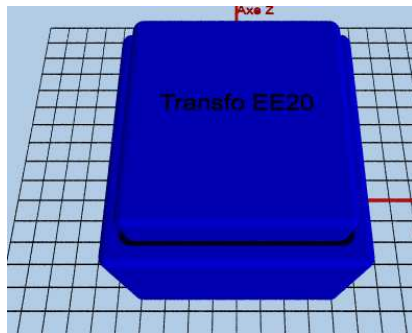
Il ne nous reste plus qu'à mettre 2 boîtes arrondies l'une sur l'autre: Une grosse dessous, une petite dessus ☺

```
#macro TRANSFO_EE20(Valeur)
union {

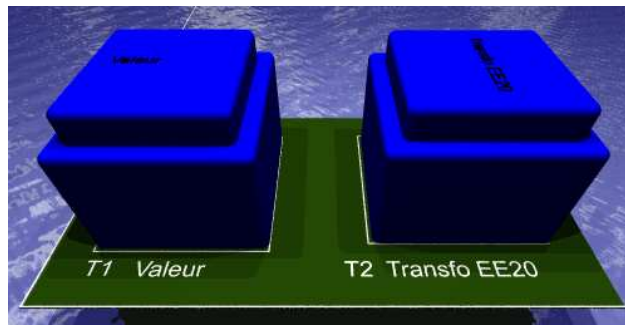
text{tff "Arial" Valeur 0.1,0 scale<2,2,1> rotate<90,0,0> translate<-6,19.1,0> pigment {Black} }

object {WT3D_BOX_ROUND_SUP_GRND (22,15,23,1) translate <0,0,0> }
object {WT3D_BOX_ROUND_SUP_GRND (18,4,19,1) translate <0,15,0> }
[...]
```

Le modèle est maintenant terminé:



Le modèle terminé (vue Visu3D)



Vue sur un typon de test

Le code complet, à ce stade, figure en annexe 4.

3c - Création de 2 macros pour la conception des broches

Nous allons écrire 2 macros pour le dessin des broches. Ces 2 macros pourront ensuite servir dans l'avenir, pour d'autres modèles.

- Une macro qui dessine une simple broche.
- Une macro qui dessine une ligne de broches: En effet, ce transfo comporte 2 lignes de 4 broches. Les broches sont souvent alignées sur les composants. CQFD !

Dessin d'une simple broche: Macro WT3D_BROCHE

```
#macro WT3D_BROCHE (posX,posZ,Hhaut,Hbas,rayon)
cylinder(<posX,Hbas,posZ> <posX,Hhaut,posZ> rayon texture {col_silver} }
#end
```

Cette macro à 4 paramètres: PosX,posZ=position X,Z Hhaut=Hauteur Y sur le circuit, Hbas=Hauteur basse, rayon=rayon de la broche

Dessin d'une ligne de broches: Macro WT3D_LIGNE_BROCHE

```
#macro WT3D_LIGNE_BROCHE (posX,posZ,Hhaut,Hbas,rayon,IncX,IncZ,Nombre)
#declare Count = 0;
#while (Count < Nombre)
WT3D_BROCHE (posX + Count * IncX,posZ + Count * IncZ,Hhaut,Hbas,rayon)
#declare Count = Count+1;
#end
#end
```

Cette macro a 8 paramètres: posX,posZ=position X,Z de la broche 1, Hhaut=Hauteur Y sur le circuit, Hbas=Hauteur basse, rayon=rayon de la broche, IncX,IncZ = incrément X et Z entre chaque broche. Nombre=Nombre de broches.

De plus, elle utilise une boucle " While " pour dessiner plusieurs broches.

Utilisons maintenant ces macros pour le modèle de notre transformateur:

```
#macro TRANSFO_EE20(Valeur)
union {
text{ttf "Arial" Valeur 0.1,0 scale<2,2,1> rotate<90,0,0> translate<-6,19.1,0> pigment {Black} }
object {WT3D_BOX_ROUND_SUP_GRND (22,15,23,1) }
object {WT3D_BOX_ROUND_SUP_GRND (18,4,19,1) translate <0,15,0> }
#if(pin_short=off)
WT3D_LIGNE_BROCHE (-7.5,-7.5,1,-8,0.25,5,0,4)
WT3D_LIGNE_BROCHE (-7.5, 7.5,1,-8,0.25,5,0,4)
#else
WT3D_LIGNE_BROCHE (-7.5,-7.5,1,-(pin_length+pcb_height),0.25,5,0,4)
WT3D_LIGNE_BROCHE (-7.5, 7.5,1,-(pin_length+pcb_height),0.25,5,0,4)
#endif
pigment {Blue}
}
#end
```

Le code complet est en annexe 5.

Ce code est maintenant bien plus court. De plus les macros créées le sont pour toujours. Sur le long terme, l'écriture et la réutilisation de macros sont fortement conseillées. Gain de temps, réutilisation de code, clarté, simplicité (mieux vaut 10 petites macros simples qu'un objet complexe !)...

Conclusion

La création d'un modèle 3D peut sembler complexe, mais elle constitue un excellent moyen de:

- Contribuer à un projet ouvert, la réalisation d'une librairie de modèles 3D
- D'apprendre la syntaxe Povray
- De découvrir et/ou approfondir le monde de la programmation (macro, paramètre...)
- De découvrir et/ou approfondir le monde de la 3D.

Vaste programme sans fin, mais passionnant !

Annexe 1 : Les recommandations pour nommer les macros

Source : www.matwei.de

Traduction : P. Boucheny : <http://perso.numericable.fr/pboucheny/eagle3d/>

- Un composant devrait uniquement être défini à l'aide d'une macro. Si vous voulez utiliser plus d'une macro, rassemblez les toutes dans une grosse macro.
- Les noms des macros sont toujours écrits en lettres capitales.
- Pour les composants qui se ressemblent les uns aux autres, une macro de base devrait être créée. L'exemple d'une telle macro de base est donné par IC_SMD_GRND(...) laquelle peut non seulement créer des boîtiers comme les SOP, SSOP, TSSOP mais aussi les SOT23 ou les DPAK.
- Les macros devraient toujours commencer avec un préfixe qui indique à quel fichier .inc elles appartiennent. Par exemple une macro qui fait partie de ic.inc devrait commencer par IC_ etc. Ce n'est pas le cas pour toutes les macros en ce moment mais cela devrait être fait pour les macros futures.
- Toutes les macros devraient commencer par un nom clair. AXBBK() n'est pas très indicatif.
- Les Macros devraient posséder des noms en anglais.

Et également :

- Les associations modèle 3D / Empreinte Wintypen seront fournies (la ligne pack).
- 1 unité Povray = 1 mm.

Annexe 2 : Options et variables

Source : www.matwei.de - adapté et modifié pour Wintypon 3D.

Traduction : P. Boucheny : <http://perso.numericable.fr/pboucheny/eagle3d/>

Réglage du fichier POVRay

Vous pouvez définir quelques réglages dans le fichier POVRay lesquels seront générés par Wintypon. Ces items sont décrits ici.

`#declare use_file_as_inc = off;` **Non utilisé par Wintypon 3D**

Régler cette valeur sur « on » pour utiliser le fichier en tant que fichier inclus dans un autre fichier POVRay. Par exemple pour générer une carte composée de plusieurs modules. La carte sera définie en tant que macro à 6 paramètres. Le nom de la macro peut être tiré du fichier POVRay après les lignes « spot ».

`#declare global_res_shape = 1;`

"0" ou "1" change la forme des résistances.

`#declare global_res_colselect = 0;`

Réglez cette valeur à « 0 » et les couleurs de la résistance seront définies par les lignes suivantes. A « 1 » la couleur est définie aléatoirement.

`#declare global_res_col = 1;`

Détermine la couleur du corps des résistances discrètes. La table des codes couleurs se trouve dans le fichier POVRay.

`#declare pcb_upsidedown = off;`

Sur « on » la carte se présente coté soudures.

`#declare pcb_rotdir = x;` **Non utilisé par Wintypon 3D**

Détermine l'axe de rotation de la carte.

`#declare environment = on;`

Permet de valider ou non les arrières-plans « vagues et nuages ».

`#declare pin_length = 2.5;`

Détermine la longueur des pattes des composants discrets.

`#declare col_preset = 2;`

Les diverses couleurs de circuit imprimé, plages, sérigraphie etc. peuvent être définies ici.

`#declare pin_short = on;`

Réglé sur « on », les pattes des composants discrets seront raccourcies à la valeur vue précédemment.

`#local pcb_parts = on;`

`#local pcb_polygons = on;`

`#local pcb_silkscreen = on;`

`#local pcb_wires = on;`

Certaines parties de la carte peuvent être inhibées ici.

`#declare pcb_layer1_used = 1;` **Non utilisé par Wintypon 3D**

`#declare pcb_layer16_used = 1;` **Non utilisé par Wintypon 3D**

`#declare pcb_height = 1.500000;` (attention anciennement `pcb_hight`)

Hauteur du circuit imprimé

`#declare pcb_cuheight = 0.035000;` (attention anciennement `pcb_cuhight`)

Epaisseur du cuivre

`#declare inc_testmode = off;`

Ne pas changer ces valeurs. **Non utilisé par Wintypon 3D**

Information: Non utilisé par Wintypon 3D: Variables utilisées par Eagle™ 3D, et pas par Wintypon 3D, mais pouvant être déclarées dans le fichier Pov généré, au cas où elles seraient utilisées dans des fichiers INC.

Annexe 3 : Macro WT3D_BOX_ROUND_SUP_GRND

```
// Boîte arrondie sur le dessus.( round box ) dimensions = TailleX, TailleY, tailleZ ( size )
// rayon : rayon de l'arrondi ( rayon: round radius )
// Wintypon 3D / Pascal EYNARD / www.typonrelais.com / 09/2006
//
#macro WT3D_BOX_ROUND_SUP_GRND (tailleX,tailleY,tailleZ,rayon)

union {

sphere {<-tailleX/2 + rayon, tailleY - rayon,-tailleZ/2 + rayon> rayon }
sphere {<-tailleX/2 + rayon, tailleY - rayon, tailleZ/2 - rayon> rayon }
sphere {< tailleX/2 - rayon, tailleY - rayon,-tailleZ/2 + rayon> rayon }
sphere {< tailleX/2 - rayon, tailleY - rayon, tailleZ/2 - rayon> rayon }

cylinder {<-tailleX/2 + rayon, tailleY - rayon,-tailleZ/2 + rayon><-tailleX/2 + rayon, tailleY/1 - rayon, tailleZ/2 - rayon> rayon }
cylinder {< tailleX/2 - rayon, tailleY - rayon, tailleZ/2 - rayon>< tailleX/2 - rayon, tailleY/1 - rayon,-tailleZ/2 + rayon> rayon }
cylinder {<-tailleX/2 + rayon, tailleY - rayon,-tailleZ/2 + rayon>< tailleX/2 - rayon, tailleY/1 - rayon,-tailleZ/2 + rayon> rayon }
cylinder {<-tailleX/2 + rayon, tailleY - rayon, tailleZ/2 - rayon>< tailleX/2 - rayon, tailleY/1 - rayon, tailleZ/2 - rayon> rayon }

cylinder {<-tailleX/2 + rayon,0 ,-tailleZ/2 + rayon><-tailleX/2 + rayon, tailleY - rayon,-tailleZ/2 + rayon> rayon }
cylinder {<-tailleX/2 + rayon,0 , tailleZ/2 - rayon><-tailleX/2 + rayon, tailleY - rayon, tailleZ/2 - rayon> rayon }
cylinder {< tailleX/2 - rayon,0 ,-tailleZ/2 + rayon>< tailleX/2 - rayon, tailleY - rayon,-tailleZ/2 + rayon> rayon }
cylinder {< tailleX/2 - rayon,0 , tailleZ/2 - rayon>< tailleX/2 - rayon, tailleY - rayon, tailleZ/2 - rayon> rayon }

box {<-tailleX/2 + rayon,0,-tailleZ/2><tailleX/2-rayon,tailleY-rayon,tailleZ/2> }
box {<-tailleX/2,0,-tailleZ/2+rayon><tailleX/2,tailleY-rayon,tailleZ/2-rayon> }
box {<-tailleX/2 + rayon,0,-tailleZ/2+rayon><tailleX/2-rayon,tailleY,tailleZ/2-rayon> }

}
#end
```

Annexe 4 : Le code (partiel) du modèle 3D du transformateur

```
//
// Fichier user.inc
// Exemple de création : Transformateur EE20
//
// Boîte arrondie sur le dessus.( round box ) dimensions = TailleX, TailleY, tailleZ ( size )
// rayon : rayon de l'arrondi ( rayon: round radius )
// Wintypou 3D / Pascal EYNARD / www.typonrelais.com / 09/2006
//
#macro WT3D_BOX_ROUND_SUP_GRND (tailleX,tailleY,tailleZ,rayon)

union {

sphere {<-tailleX/2 + rayon, tailleY - rayon,-tailleZ/2 + rayon> rayon }
sphere {<-tailleX/2 + rayon, tailleY - rayon, tailleZ/2 - rayon> rayon }
sphere {< tailleX/2 - rayon, tailleY - rayon,-tailleZ/2 + rayon> rayon }
sphere {< tailleX/2 - rayon, tailleY - rayon, tailleZ/2 - rayon> rayon }

cylinder {<-tailleX/2 + rayon, tailleY - rayon,-tailleZ/2 + rayon><-tailleX/2 + rayon, tailleY/1 - rayon, tailleZ/2 - rayon> rayon }
cylinder {< tailleX/2 - rayon, tailleY - rayon, tailleZ/2 - rayon>< tailleX/2 - rayon, tailleY/1 - rayon,-tailleZ/2 + rayon> rayon }
cylinder {<-tailleX/2 + rayon, tailleY - rayon,-tailleZ/2 + rayon>< tailleX/2 - rayon, tailleY/1 - rayon,-tailleZ/2 + rayon> rayon }
cylinder {<-tailleX/2 + rayon, tailleY - rayon, tailleZ/2 - rayon>< tailleX/2 - rayon, tailleY/1 - rayon, tailleZ/2 - rayon> rayon }

cylinder {<-tailleX/2 + rayon,0 ,-tailleZ/2 + rayon><-tailleX/2 + rayon, tailleY - rayon,-tailleZ/2 + rayon> rayon }
cylinder {<-tailleX/2 + rayon,0 , tailleZ/2 - rayon><-tailleX/2 + rayon, tailleY - rayon, tailleZ/2 - rayon> rayon }
cylinder {< tailleX/2 - rayon,0 ,-tailleZ/2 + rayon>< tailleX/2 - rayon, tailleY - rayon,-tailleZ/2 + rayon> rayon }
cylinder {< tailleX/2 - rayon,0 , tailleZ/2 - rayon>< tailleX/2 - rayon, tailleY - rayon, tailleZ/2 - rayon> rayon }

box {<-tailleX/2 + rayon,0,-tailleZ/2><tailleX/2-rayon,tailleY-rayon,tailleZ/2> }
box {<-tailleX/2,0,-tailleZ/2+rayon><tailleX/2,tailleY-rayon,tailleZ/2-rayon> }
box {<-tailleX/2 + rayon,0,-tailleZ/2+rayon><tailleX/2-rayon,tailleY,tailleZ/2-rayon> }

}
#end

// Transformateur type EE20 / 23 * 22 mm / hauteur = 19mm
// Wintypou 3D / Pascal EYNARD / www.typonrelais.com / 09/2006
//
#macro TRANSFO_EE20(Valeur)
union {

text{tff "Arial" Valeur 0.1,0 scale<2,2,1> rotate<90,0,0> translate<-6,19.1,0> pigment {Black} }

object {WT3D_BOX_ROUND_SUP_GRND (22,15,23,1) }
object {WT3D_BOX_ROUND_SUP_GRND (18,4,19,1) translate <0,15,0> }

#if(pin_short=off)
cylinder{<-7.5,-8,-7.5><-7.5,1,-7.5>0.25 texture {col_silver} }
cylinder{<-2.5,-8,-7.5><-2.5,1,-7.5>0.25 texture {col_silver} }
cylinder{<2.5,-8,-7.5><2.5,1,-7.5>0.25 texture {col_silver} }
cylinder{<7.5,-8,-7.5><7.5,1,-7.5>0.25 texture {col_silver} }
cylinder{<-7.5,-8,7.5><-7.5,1,7.5>0.25 texture {col_silver} }
cylinder{<-2.5,-8,7.5><-2.5,1,7.5>0.25 texture {col_silver} }
cylinder{<2.5,-8,7.5><2.5,1,7.5>0.25 texture {col_silver} }
cylinder{<7.5,-8,7.5><7.5,1,7.5>0.25 texture {col_silver} }

#else

cylinder{<-7.5,-(pin_length+pcb_height),-7.5><-7.5,1,-7.5>0.25 texture {col_silver} }
cylinder{<-2.5,-(pin_length+pcb_height),-7.5><-2.5,1,-7.5>0.25 texture {col_silver} }
cylinder{<2.5,-(pin_length+pcb_height),-7.5><2.5,1,-7.5>0.25 texture {col_silver} }
cylinder{<7.5,-(pin_length+pcb_height),-7.5><7.5,1,-7.5>0.25 texture {col_silver} }
cylinder{<-7.5,-(pin_length+pcb_height),7.5><-7.5,1,7.5>0.25 texture {col_silver} }
cylinder{<-2.5,-(pin_length+pcb_height),7.5><-2.5,1,7.5>0.25 texture {col_silver} }
cylinder{<2.5,-(pin_length+pcb_height),7.5><2.5,1,7.5>0.25 texture {col_silver} }
cylinder{<7.5,-(pin_length+pcb_height),7.5><7.5,1,7.5>0.25 texture {col_silver} }

#endif

pigment {Blue}
}
#end
```

Annexe 5 : Le code complet du modèle 3D du transformateur

```
// Boîte arrondie sur le dessus.( top round box )
// dimensions = TailleX, TailleY, tailleZ ( size )
// rayon : rayon de l'arrondi ( rayon: round radius )
//
// Wintypon 3D / Pascal EYNARD / www.typonrelais.com / 09-2006
// FR: Licence : Créative Commons : Paternité - Pas d'Utilisation Commerciale 2.0 France // http://creativecommons.org/licenses/by-nc/2.0/fr/
// EN: Licence : Creative Commons : Attribution - Non Commercial 2.0 France // http://creativecommons.org/licenses/by-nc/2.0/fr/deed.en
//
#macro WT3D_BOX_ROUND_SUP_GRND (tailleX,tailleY,tailleZ,rayon)
union {
sphere {<-tailleX/2 + rayon, tailleY - rayon,-tailleZ/2 + rayon> rayon }
sphere {<-tailleX/2 + rayon, tailleY - rayon, tailleZ/2 - rayon> rayon }
sphere {< tailleX/2 - rayon, tailleY - rayon,-tailleZ/2 + rayon> rayon }
sphere {< tailleX/2 - rayon, tailleY - rayon, tailleZ/2 - rayon> rayon }
cylinder {<-tailleX/2 + rayon, tailleY - rayon,-tailleZ/2 + rayon><-tailleX/2 + rayon, tailleY/1 - rayon, tailleZ/2 - rayon> rayon }
cylinder {< tailleX/2 - rayon, tailleY - rayon, tailleZ/2 - rayon>< tailleX/2 - rayon, tailleY/1 - rayon,-tailleZ/2 + rayon> rayon }
cylinder {<-tailleX/2 + rayon, tailleY - rayon,-tailleZ/2 + rayon>< tailleX/2 - rayon, tailleY/1 - rayon,-tailleZ/2 + rayon> rayon }
cylinder {<-tailleX/2 + rayon, tailleY - rayon, tailleZ/2 - rayon>< tailleX/2 - rayon, tailleY/1 - rayon, tailleZ/2 - rayon> rayon }
cylinder {<-tailleX/2 + rayon,0 , tailleZ/2 - rayon><-tailleX/2 + rayon, tailleY - rayon,-tailleZ/2 + rayon> rayon }
cylinder {< tailleX/2 - rayon,0 , -tailleZ/2 + rayon>< tailleX/2 - rayon, tailleY - rayon,-tailleZ/2 + rayon> rayon }
cylinder {< tailleX/2 - rayon,0 , tailleZ/2 - rayon>< tailleX/2 - rayon, tailleY - rayon, tailleZ/2 - rayon> rayon }
box {<-tailleX/2 + rayon,0,-tailleZ/2><tailleX/2-rayon,tailleY-rayon,tailleZ/2> }
box {<-tailleX/2,0,-tailleZ/2+rayon><tailleX/2,tailleY-rayon,tailleZ/2-rayon> }
box {<-tailleX/2 + rayon,0,-tailleZ/2+rayon><tailleX/2-rayon,tailleY,tailleZ/2-rayon> }
}
#end

// Broche verticale simple ( vertically pin )
// PosX,posZ=position X,Z Hhaut=Hauteur Y sur le circuit, Hbas=Hauteur basse, rayon=rayon de la broche
// ( X position,Y position, Top height, Bottom height,radius )
//
// Wintypon 3D / Pascal EYNARD / www.typonrelais.com / 09-2006
// FR: Licence : Créative Commons : Paternité - Pas d'Utilisation Commerciale 2.0 France // http://creativecommons.org/licenses/by-nc/2.0/fr/
// EN: Licence : Creative Commons : Attribution - Non Commercial 2.0 France // http://creativecommons.org/licenses/by-nc/2.0/fr/deed.en
//
#macro WT3D_BROCHE (posX,posZ,Hhaut,Hbas,rayon)
cylinder{<posX,Hbas,posZ> <posX,Hhaut,posZ> rayon texture {col_silver} }
#end

// Ligne de broches ( Line of pin )
// posX,posZ=position X,Z de la broche 1
// Hhaut=Hauteur Y sur le circuit, Hbas=Hauteur basse, rayon=rayon de la broche
// IncX,IncZ = increment X et Z entre chaque broche. Nombre=Nombre de broche
// (X position,Z position,Top height,Bottom height,radius,X increment, Z increment, nombre)
//
// Wintypon 3D / Pascal EYNARD / www.typonrelais.com / 09-2006
// FR: Licence : Créative Commons : Paternité - Pas d'Utilisation Commerciale 2.0 France // http://creativecommons.org/licenses/by-nc/2.0/fr/
// EN: Licence : Creative Commons : Attribution - Non Commercial 2.0 France // http://creativecommons.org/licenses/by-nc/2.0/fr/deed.en
//
#macro WT3D_LIGNE_BROCHE (posX,posZ,Hhaut,Hbas,rayon,IncX,IncZ,Nombre)
#declare Count = 0;
#while (Count < Nombre)
WT3D_BROCHE (posX + Count * IncX,posZ + Count * IncZ,Hhaut,Hbas,rayon)
#declare Count = Count+1;
#end
#end

// Transformateur type EE20 / 23 * 22 mm / hauteur = 19mm
//
// Wintypon 3D / Pascal EYNARD / www.typonrelais.com / 09-2006
// FR: Licence : Créative Commons : Paternité - Pas d'Utilisation Commerciale 2.0 France // http://creativecommons.org/licenses/by-nc/2.0/fr/
// EN: Licence : Creative Commons : Attribution - Non Commercial 2.0 France // http://creativecommons.org/licenses/by-nc/2.0/fr/deed.en
//
#macro TRANSFO_EE20(Valeur)
union {
text{ttf "Arial" Valeur 0.1,0 scale<2,2,1> rotate<90,0,0> translate<-6,19.1,0> pigment {Black} }
object {WT3D_BOX_ROUND_SUP_GRND (22,15,23,1) }
object {WT3D_BOX_ROUND_SUP_GRND (18,4,19,1) translate <0,15,0> }
#if(pin_short=off)
WT3D_LIGNE_BROCHE (-7.5,-7.5,1,-8,0.25,5,0,4)
WT3D_LIGNE_BROCHE (-7.5, 7.5,1,-8,0.25,5,0,4)
#else
WT3D_LIGNE_BROCHE (-7.5,-7.5,1,-(pin_length+pcb_height),0.25,5,0,4)
WT3D_LIGNE_BROCHE (-7.5, 7.5,1,-(pin_length+pcb_height),0.25,5,0,4)
#end
pigment {Blue}
}
#end
```

Annexe 6 : Liens Internet

www.typonrelais.com

Logiciel Wintypon

sur PovRay

www.povray.org

<http://pov.monde.free.fr/>

<http://perso.orange.fr/sebastien.bancquart>

<http://www.lightning-generator.org/>

<http://povwiki.ovh.org/>

<http://www.3dvh.com>

Site principal (en anglais)

Site francophone - Aide de povray traduite - liens - Docs...

Didacticiel complet en français

Site francophone très riche - Docs, liens..

Un wiki sur Povray, très très riche.

Portail français sur la 3D en général

Modèles de composants 3D

www.matwei.de

<http://perso.numericable.fr/pboucheny/eagle3d/>

http://www.adrirobot.it/menu_new/index/index_eagle.htm

Modèle de composants 3D

Modèle de composants 3D + didacticiels

Modèle de composants 3D (Afficheurs...)

<http://www.geocities.com/cdavidrp/PovRay/TutorialCreacionDeComponentesElectronicosEn3D.pdf>

Création d'un afficheur (langue espagnol)

http://www.f-lohmueller.de/pov_tut/x_sam/tec_010e.htm

Tutorial Création afficheur

<http://www.todopic.com.ar/foros/index.php?topic=26048.0>

Création de composants et autres

Avancé

<http://www.ignorancia.org/es/index.php?page=eagle3d>

Augmentation du réalisme (en Espagnol)

<http://www.ucontrol.com.ar/forosmf/programacion-en-visual-basic/tutorial-pov-ray-desde-cero-%28orientado-a-la-electronica%29/>