# 比較対照 METAPOST & Asymptote 〜 第4回

比較対照と言いながら，あんまり比較してないかも。前回は Asymptote 独特の話だったけど前々回についてちょっと補足。clip は Asymptote と METAPOST で大分使い方が違って，METAPOST では

clip〈ピクチャー〉to〈巡回パス〉;

で，〈ピクチャー〉[1] の〈巡回パス〉からはみ出した部分を取り去ります。

では，今回のお題。今回は前回説明できなかった，図形（〈ピクチャー〉）の貼り付けについて。

## 1 picture 値変数の使い方

**picture** 値変数は，絵を描くための「カンバス」のようなものです。通常「カンバス」は 1 枚あれば十分なのですが，clip のように図全体にかかる命令を実行するとき，「図のこの部分は clip したいけど，ここはしたくない」ということもあるでしょう。こういう場合，「カンバス」をもう一枚用意して clip される図形を描き，clip を実行し，それをもとの「カンバス」に貼り付ける，という操作が考えられます。こういった場合，**picture** 値変数は役に立ちます。

なお，普段 **picture** 値変数を指定せずに draw や fill を実行していますが，これは currentpicture という **picture** 値変数が初めから定義されていて，Asymptote では，とくに描画先の **picture** 値変数が指定されないとき，デフォルトで currentpicture が使われているためです[2]。

たとえば下の図で，円の重なっている部分が別の色になっていますが，PostScript では色は上書き方式ですので，重ね塗りしても色が勝手に混ざってくれたりはしません。この図では，円で区切られた領域をそれぞれ別々に塗っています。ここで，既存の〈巡回パス〉で囲まれた領域から，重なった部分の境界線を作り出す必要が出てきますが，これは割と面倒です[3]。「重なった部分」を描くだけなら，まず 1 つの円を塗りつぶして，他の円の外部にあたる部分を clip すればいいのですが，それではそれまでに描いた図も clip されてしまいます。そこで，別個に **picture** 値変数を用意して，そこに「重なった部分」を描いてから，currentpicture に貼り付けることを考えます。

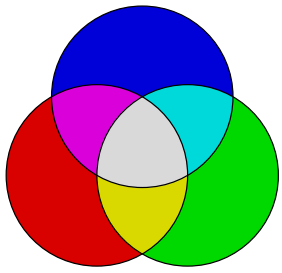
まず，新たに（たとえば pic という名前の）**picture** 値変数を用意します。やり方は **pair** 値のときなどと同様で，

```
picture pic = new picture;
```

です。その上で，draw や fill のオプション引数として pic を指定すれば，pic 上への描画になります。そして，このようにして **picture** 値変数 pic に描かれた図を currentpicture に貼り付けるのが，add という命令です。

では，具体例を見てみましょう[4]。

```
\begin{ASYpic}<1.2cm>(0,0)(2,-2)
  \sendASY{picture pic = new picture;
    path[] pth = {circle(0,1), circle(right,1), circle(dir(60),1),circle(0,1)};
    pen[] pn = {0.85red,0.85green,0.85blue,0.85red};
    for(int i=0; i<3; ++i){fill(pth[i],pn[i]);}
    for(int i=0; i<3; ++i){fill(pic,pth[i],pn[i]+pn[i+1]);
      clip(pic,pth[i+1]);  add(pic);  erase(pic);}
    fill(pic,pth[0],pn[0]+pn[1]+pn[2]);
    clip(pic,pth[1]);  clip(pic,pth[2]);  add(pic);
    for(int i=0; i<3; ++i){draw(pth[i]);}
  }
\end{ASYpic}
```



この例では，色については加算，定数倍が有効であることも示しています。**pen** 値の加算で，同種のものを同時に指定した場合，先に書かれた方は無視され，後の方のもののみ有効になります。たとえば線種に関して実線（solid）と

[1] 現在描画中の図の場合，ここは currentpicture です。

[2] METAPOST では，そもそも draw や fill が currentpicture への描画命令として定義されています（つまり currentpicture 専用）。MEPOTEX では，一般の **picture** 値変数に書き込める draw や fill の拡張命令 drawpic , fillpic が定義されています。

[3] METAPOST にも Asymptote にも buildcycle という命令があり，これを使うと比較的楽に既存の〈パス〉から新しい〈巡回パス〉を作れますが，元の〈パス〉が〈巡回パス〉だと，〈パス〉と〈パス〉の交点が 2 個以上あるため，どの交点とどの交点をどういう順番で結ぶかによって囲まれる部分が変わってきます。このあたりは，うちのサイトの METAPOST ページにある「METAFONT & METAPOST で楽々お絵かき」の第 2 部第 6 講に詳しく書いてありますので，そちらを御覧下さい。

[4] circle は中心（**pair** 値）と半径（**real** 値）を指定し，円を描く命令です。なお，**pair** 値が要求されているところで **real** 値（たとえば t）を指定すると，**pair** 値 (t,0) が指定されたものと解釈されますので，原点は 0 で OK です。

破線 ( dashed ) を solid+dashed のように指定した場合，solid は無視され dashed のみが有効になります。しかし，色だけは別で，例に挙げたように red ( rgb 値で (1,0,0) ) と green ( rgb 値で (0,1,0) ) を加算すれば yellow ( rgb 値で (1,1,0) ) になります[5]。

ちなみに，draw や fill のオプション引数としては，これまで，色・線種などを指定する **pen** 値，固定サイズで貼り付けるときの貼り付け位置を指定する **pair** 値があることを説明しましたが，今回の **picture** 値も含めて，その順序を整理しておくと，次のようになります[6]。

fill(**pair** 値引数，**picture** 値引数，**path** 値引数 (必須)，**pen** 値引数)　　( draw も同様。)

## 2　貼り付ける図のサイズ

第 2 回で，Asymptote は **picture** 変数毎に別々の size あるいは unitsize を持てるといいましたが，そのことについてもう少し詳しく説明します。

第 2 回で触れたとおり，たとえば unitsize(2); とすれば，最終的な図のサイズが 2 倍され，単位長が 2 倍されたのと同じ効果を生みます。これは正確には currentpicture が 2 倍に拡大されているのです。size , unitsize にはオプション引数があり，currentpicture 以外の **picture** 値変数の拡大・縮小を行うこともできます。たとえば，**picture** 値変数 pic に描かれた ( あるいはこれから描かれる ) 図を 0.5 倍に縮小したいなら，unitsize(pic,0.5); とします。さらに，size , unitsize では横方向と縦方向に別々の倍率を指定することもできます。たとえば，先の例で横方向を 0.5 倍，縦方向を 1.5 倍にしたいなら unitsize(pic,0.5,1.5); とします。

こうして描かれた図を，add によって currentpicture に貼り付けるとき，貼り付け方によって，拡大率の扱いが変わってきます。以下，それを順番に説明していきます。

♠ まずは，固定サイズで貼り付け。

add では，貼り付ける画像の**後**に[7] pair 値のオプション引数をとることができ，この引数が指定されたときは，貼り付ける画像に指定された size や unitsize , 貼り付け先 ( デフォルトは currentpicture ) に指定された size や unitsize は一切無視され，元々の図のサイズで貼り付けられます。ただし，貼り付け位置を指定する **pair** 値は，貼り付け先の座標での値と解釈されます。

♠ 次は，貼り付ける図の拡大率を反映させる方法。

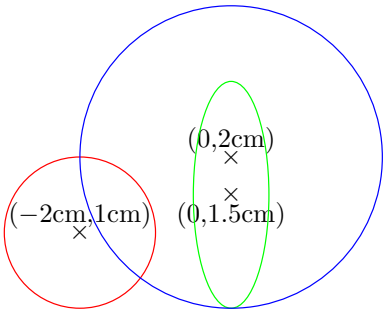
これはたとえば，**picture** 値変数 pic を貼り付ける場合，add(pic.fit()); のように貼り付ける画像に .fit() をつけて指定します[8]。なお，.fit() は上記「固定サイズ」より優先されますので，add(pic.fit(),(1cm,0)); とすると，pic の拡大率を反映させた上で，currentpicture の拡大率 × (1cm,0) の位置に図が貼り付けられます。

♠ 最後に，貼り付け先の拡大率に合わせる方法。

これはたとえば，**picture** 値変数 pic を貼り付ける場合，単に add(pic); とすれば実現できます。

実例を挙げましょう[9]。

```
\begin{ASYpic}*<1.2cm>(0,0)(2,-2)
  \sendASY{picture pic = new picture;
    unitsize(pic,0.5,1.5); unitsize(2);
    draw(pic,circle((0,1cm),1cm),red);   add(pic,(-1cm,0)); erase(pic);
    draw(pic,circle((0,1cm),1cm),green);  add(pic.fit());  erase(pic);
    draw(pic,circle((0,1cm),1cm),blue);   add(pic);
  }
  \astput(-2,1) {$\times$}  \astput(-2,1)[b]<0mm,0.5mm>{($-2$cm,1cm)}
  \astput( 0,1.5){$\times$} \astput(0,1.5)[t]<0mm,-1mm>{(0,1.5cm)}
  \astput( 0,2) {$\times$}  \astput(0,2)[b]<0mm,0.5mm>{(0,2cm)}
\end{ASYpic}
```




[5] METAPOST では色は rgb 値で指定します。MEPOTEX では cmyk や hsb 形式も受け付けるようマクロを組んでいますが，内部では結局 rgb に換算して管理していますので，加算は rgb 形式での加算になります。それに対して Asymptote は cmyk 形式での加算もサポートしていますから，たとえば，Magenta ( cmyk 値で (0,1,0,0) ) と Yellow ( cmyk 値で (0,0,1,0) ) を加算すれば cmyk 値 (0,1,1,0) ( 赤 ) になります。小文字の yellow は rgb 値，大文字始まりの Yellow は cmyk 値で定義されていることにも注意してください。( Red はデフォルトでは定義されていません。)

[6] 実はまだまだオプション引数があるのですが，それは追々。add のオプション引数も多数あるのですが，それも追々。

[7] 同様の機能を持つ draw や fill のオプション引数は，引数の先頭に指定するので混乱しないようにしてください。

[8] fit には引数があるのですが，引数の説明にはかなり内部的なことを説明する必要がありますので，ここではデフォルト値を使うことにします。( 必須の引数はないので，引数は「空」でかまいません。)

[9] add(pic,(-1cm,0)) では pic は固定サイズで倍率 1 倍，currentpicture は倍率 2 倍なので，中心は 1*(0,1cm)+2*(-1cm,0) です。