The Isometry Cometh

Transformations that alter the position but not the shape of an object are known as isometries. Four basic types of isometric transformation exist: translation, rotation, reflection and glide reflection. Translation is a simple 'slide' of the original figure. Rotation turns the shape about some specified central point. Reflection involves the movement of points across a mirror line so that each point in the final shape lies the same distance on one side of the line as the corresponding point in the original did on the other. Glide reflection is a combination of a reflection and a translation. Whilst translation and rotation preserve 'sense', reflection and glide reflection change it: imagine reflecting a word in a mirror, for example

```
TO REWRITE :PROC
   OUTPUT REWRITE.PROC TEXT :PROC
END
```

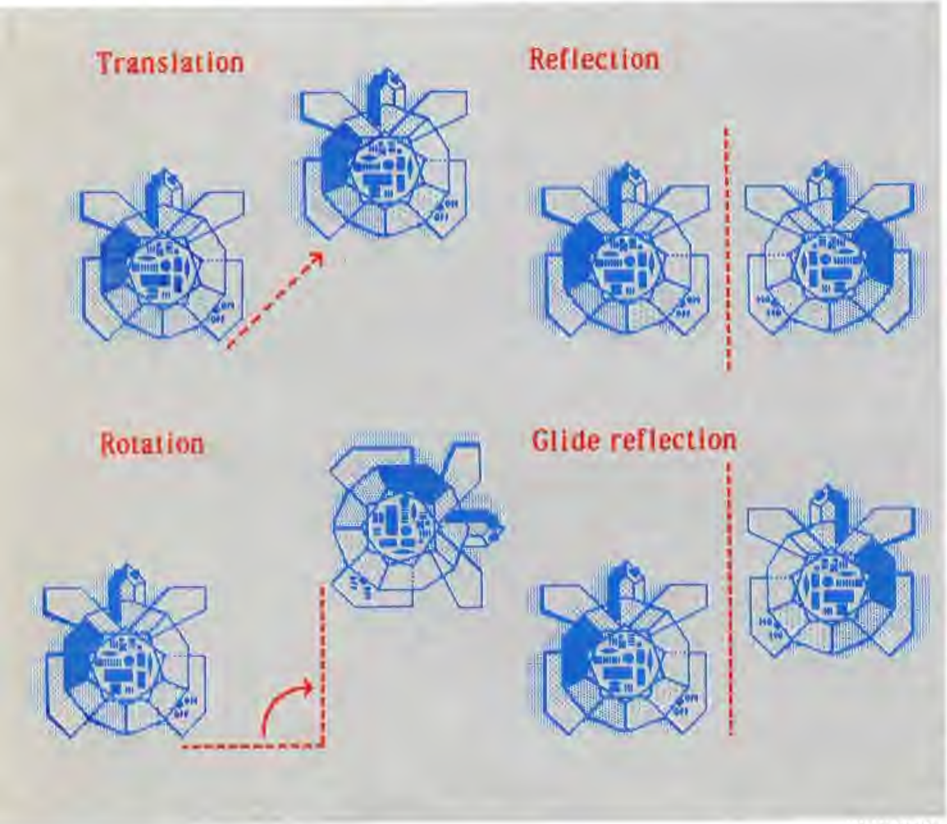
REWRITE takes the text of a specified procedure, alters it and outputs it under another name. It assumes that the procedure it is working on is written in terms of LOGO primitives and does not contain any subprocedures. REWRITE contains a call to the following procedures:

```
TO REWRITE.PROC :TEXT
   IF :TEXT = [] THEN OUTPUT []
   OUTPUT FPUT REWRITE.LINE FIRST :TEXT
   REWRITE. PROC BUTFIRST :TEXT
END
```

This procedure divides the task of rewriting the input procedure into individual lines, by calling the following procedure:

```
TO REWRITE.LINE :LINE
   IF :LINE = [] THEN OUTPUT []
   IF LIST? FIRST :LINE THEN OUTPUT FPUT
   REWRITE.LINE FIRST : LINE REWRITE.LINE
   BUTFIRST :LINE
   OUTPUT FPUT
   CHANGE.WORD FIRST :LINE
   REWRITE.LINE BUTFIRST :LINE
END
```

REWRITE.LINE does the processing on each line, passing individual words on to CHANGE.WORD for it to deal with. The line beginning IF LIST? is needed in order to deal with a situation where MOTIF contains a REPEAT statement. If you exclude this possibility in your MOTIF procedures, then you can remove the line from this procedure. The listing for CHANGE.WORD is:



IAN McKINNELL

```
TO CHANGE.WORD :WORD
   IF ( ANYOF :WORD = "RT :WORD = "RIGHT ) THEN
   OUTPUT "LEFT
   IF (ANYOF :WORD = "LT :WORD = "LEFT ) THEN
   OUTPUT "RIGHT
   OUTPUT :WORD
END
```

This procedure checks each individual word and makes any necessary alterations. Having entered all these procedures, let's see how they work. First of all, we need to define a simple shape, such as:

```
TO TRI
   REPEAT 3 [FD 50 RT 120]
END
```

Now, enter DEFINE "REF REWRITE "TRI, and call up REF. Its definition should be:

```
TO REF
   REPEAT 3 [FD 50 LEFT 120]
END
```

It is quite possible to write a more general REWRITE procedure that will also rewrite any subprocedures called by the main procedure. If you should try to write this, take care with recursive procedures! You'll also need to be able to test whether a word is a procedure name.

THE SEVEN STRIP PATTERNS

It would be possible (and mathematically elegant) to build up the patterns from procedures for the four basic transformations. The pattern-drawing procedures make use of three helping subprocedures. These are:

```
TO POSITION
   HT
   PU
   SETXY – 125 0
   PD
END
```

This positions the turtle at the left-hand side of the screen, ready to begin drawing.

```
TO MOVE
   PU
   RT 90
   FD 50
   LT 90
   PD
END
```

MOVE performs the required translation.

```
TO TURN
   RT 180
END
```

TURN performs the one rotation that we require.

To use these procedures first define a shape procedure (say, SHAPE) which is state transparent and has no subprocedure calls. Then you can draw the first pattern using SHAPE as your motif by entering PATTERN1 "SHAPE.