The same transformer from part (a) works, but for this direction, it is more conveniently written.

(c) What is the data transformer to transform from polar to Cartesian co-ordinates?

\[ x^2 + y^2 = r^2 \wedge \sin a = y/r \wedge \cos a = x/r \wedge \tan a = y/x \]

This transform has some redundancy; any two of those conjuncts imply the other two. We can already see a constraint on its use: \( r \neq 0 \wedge x \neq 0 \). This constraint has some redundancy: if \( x = 0 \) then \( r \neq 0 \). To transform from Cartesian to polar, this transform is more conveniently written

\[ x = r \times \cos a \wedge y = r \times \sin a \]

to use one-point laws to get rid of quantifications \( \forall x, y \) and \( \exists x', y' \).

(b) In Cartesian co-ordinates, one of the operations on a pixel is \textit{translate}, which moves a pixel from position \( x \) and \( y \) to position \( x+u \) and \( y+v \).

\[
\text{translate} = x := x+u \wedge y := y+v
\]

Use the data transformer from (a) to transform operation \textit{translate} from Cartesian to polar co-ordinates.

\[
\forall x, y \quad x = r \times \cos a \wedge y = r \times \sin a
\]

\[
\Rightarrow \exists x', y' \quad x' = r' \times \cos a' \wedge y' = r' \times \sin a' \wedge \text{translate}
\]

\[
\forall x, y \quad x = r \times \cos a \wedge y = r \times \sin a
\]

\[
\Rightarrow \exists x', y' \quad x' = r' \times \cos a' \wedge y' = r' \times \sin a' \wedge (x := x+u. \ y := y+v)
\]

\[
\forall x, y \quad x = r \times \cos a \wedge y = r \times \sin a
\]

\[
\Rightarrow \exists x', y' \quad x' = r' \times \cos a' \wedge y' = r' \times \sin a' \wedge x := x+u \wedge y := y+v
\]

one-point \( x' \ y' \)

\[
\forall x, y \quad x = r \times \cos a \wedge y = r \times \sin a
\]

\[
\Rightarrow r' \times \cos a' = x+u \wedge r' \times \sin a' = y+v
\]

one-point \( x \ y \)

\[
\exists x', y' \quad x' = r \times \cos a \wedge y' = r \times \sin a \wedge r' \times \cos a' = (r \times \cos a)+u \wedge r' \times \sin a' = (r \times \sin a)+v
\]

This is not yet a program. It appears that the way to get a \textit{translate} program in polar co-ordinates is to transform to Cartesian, \textit{translate} in Cartesian, then transform back to polar.

\[
\exists x, y \quad \text{real} \ x := (r \times \cos a)+u \wedge y := (r \times \sin a)+v.
\]

\[
r := (x^2+y^2)^{1/2} \wedge a := \arctan(y/x)
\]

(c) What is the data transformer to transform from polar to Cartesian co-ordinates?

The same transformer from part (a) works, but for this direction, it is more conveniently rewritten, using trigonometric identities, as

\[
r = (x^2+y^2)^{1/2} \wedge a = \arctan(y/x)
\]

to use one-point laws to get rid of quantifications \( \forall r, a \) and \( \exists r', a' \).

(d) In polar co-ordinates, one of the operations on a pixel is \textit{rotate} by \( d \) radians.

\[
\text{rotate} = a := a+d
\]

Use the data transformer from (c) to transform operation \textit{rotate}.

\[
\forall r, a \quad r = (x^2+y^2)^{1/2} \wedge a = \arctan(y/x)
\]

\[
\Rightarrow \exists r', a' \quad r' = (x'^2+y'^2)^{1/2} \wedge a' = \arctan(y'/x') \wedge \text{rotate}
\]

\[
\forall r, a \quad r = (x^2+y^2)^{1/2} \wedge a = \arctan(y/x)
\]

\[
\Rightarrow \exists r', a' \quad r' = (x'^2+y'^2)^{1/2} \wedge a' = \arctan(y'/x') \wedge (a := a+d)
\]

\[
\forall r, a \quad r = (x^2+y^2)^{1/2} \wedge a = \arctan(y/x)
\]

one-point \( r' \ a' \)

\[
\forall r, a \quad r = (x^2+y^2)^{1/2} \wedge a = \arctan(y/x)
\]
\[
(x'^2+y'^2)^{1/2} = r \land \arctan(y'/x') = a+d
\]

\[
(x'^2+y'^2)^{1/2} = (x^2+y^2)^{1/2} \land \arctan(y'/x') = \arctan(y/x) + d
\]

This is not yet a program. It appears that the way to get a \textit{rotate} program in Cartesian
cordinates is to transform to polar, \textit{rotate} in polar, then transform back to Cartesian.

\[
\text{var } r, a: \text{real}; \quad r := (x^2+y^2)^{1/2}. \quad a := \arctan(y/x) + d.
\]
\[
x := r \times \cos a. \quad y := r \times \sin a
\]