where expectation is with respect to $\tilde{\mathbf{w}}(t) = \mathbf{w}(t+1)$, and note that

$$\begin{aligned} V_t(\mathbf{x},\mathbf{y}) &= \max_{\mathbf{u}\in U_t(\mathbf{x})} E\left[g_t(\mathbf{x},\mathbf{u},\mathbf{y}) + V_{t+1}(\mathbf{f}_t(\mathbf{x},\mathbf{u},\mathbf{y}),\tilde{\mathbf{w}}(t))\right] \\ &= \max_{\mathbf{u}\in U_t(\mathbf{x})} \{g_t(\mathbf{x},\mathbf{u},\mathbf{y}) + E[V_{t+1}(\mathbf{f}_t(\mathbf{x},\mathbf{u},\mathbf{y}),\tilde{\mathbf{w}}(t))]\} \\ &= \max_{\mathbf{u}\in U_t(\mathbf{x})} \{g_t(\mathbf{x},\mathbf{u},\mathbf{y}) + G_{t+1}(\mathbf{f}_t(\mathbf{x},\mathbf{u},\mathbf{y}))\}. \end{aligned}$$

Now substituting $\mathbf{y} = \mathbf{w}(t)$ and taking expectations with respect to $\mathbf{w}(t)$ on both sides above we obtain

$$G_t(\mathbf{x}) = E\left[\max_{\mathbf{u}\in U_t(\mathbf{x})} \{g_t(\mathbf{x},\mathbf{u},\mathbf{w}(t)) + G_{t+1}(\mathbf{f}_t(\mathbf{x},\mathbf{u},\mathbf{w}(t))\}\right],$$

which gives us a recursion exactly of the form (D.3).

Note, however, that by using this transformation we have reduced the original dynamic programming recursion from one with a state space $S_t \times W_t$ to one with only a state space of S_t. The function $G_t(\mathbf{x})$ has a similar interpretation as $V_t(\mathbf{x},\mathbf{y})$ for this reduced state—namely, it is the optimal expected reward-to-go from time t onward given we are in the reduced state $\mathbf{x}(t)$ at time t, where $\mathbf{y} = \mathbf{w}(t)$ still uncertain (recall $G_t(\mathbf{x}) = E[V_t(\mathbf{x},\mathbf{w}(t))]$). Indeed, one can think of this new recursion as propagating the system in two stages: first, the state $\mathbf{x}$ is realized but $\mathbf{y}$ remains uncertain. We measure the optimal expected reward at this point, yielding $G_t(\mathbf{x})$. Then the value $\mathbf{y} = \mathbf{w}(t)$ is realized, and we make our optimal decision. This takes us to a new state $\mathbf{x}(t+1)$, and the process repeats. Finally, note that this reduced-form recursion results in an optimization step of the form E[max{ }] rather than the max E[{ }] found in traditional dynamic programming formulations.

Here's a typical example of how this transformation arises in RM. Suppose $x(t)$ is a scalar capacity, $y(t)$ is the revenue of the request in period t, and $u(t) = 1$ if we decide to accept a request and zero otherwise. So the reward function is simply

$$y(t)u(t).$$

Capacity evolves according to the system equation

$$x(t+1) = x(t) - u(t),$$

and the revenue is driven by a random process

$$y(t+1) = w(t+1).$$

Formulated in traditional terms, we obtain

$$V_t(x,y) = \max_{u\in\{0,1\}} E\left[yu + V_{t+1}(x-u, w(t))\right].$$

However, with the transformation above, we can rewrite this in observable-disturbance form as

$$G_t(x) = E\left[\max_{u\in\{0,1\}} \{w(t)u + G_{t+1}(x-u)\}\right].$$

Since most dynamic programs in RM are of this observable-disturbance form, we typically use the simpler E[max{ }] rather than the traditional max E[{ }] form.