Slide 1: Title (about 0:40)

Good morning, and thank you for the introduction. My name is Kenta Kasai from the Institute of Science Tokyo. Today I will talk about breaking the orthogonality barrier in quantum LDPC codes, with a focus on a concrete construction and decoding results. To ensure accuracy and proper phrasing, I will read from a prepared script. **[Next slide.]**

Slide 2: Motivation: Classical vs Quantum LDPC (about 1:20)

In classical coding, sparse random graphs and large girth give us strong BP performance and large minimum distance. Density evolution tells us BP performance is highly sensitive to degree distributions. For regular degree at least three, the minimum distance can scale linearly with block length. So it is natural to ask if we can transfer this recipe to quantum codes. The obstacle is the CSS orthogonal constraint which couples the two matrices. Here a prime $(')$ denotes transpose. If we enforce orthogonality and regularity naively, we often get short cycles and small distance bounds. Our goal is to keep classical LDPC structure without paying that price. **[Next slide.]**

Slide 3: Problem: Row Removal Creates Logicals (about 1:30)

We start from a parent construction, abstracted from Hagiwara–Imai codes and bicycle codes, which makes degree distributions easy to control. In this construction, we first use the block-circulant structure to build fully orthogonal parent matrices $\hat{H}_X$ and $\hat{H}_Z$ with zero coding rate. To increase the rate, we remove some rows and keep only the active parts H_X and H_Z. The removed rows are the latent matrices, written $\tilde{H}_X, \tilde{H}_Z$

(i.e., $\tilde{H}$). But full orthogonality then forces the active–latent blocks to be orthogonal as well. So removed rows can sit inside the code spaces and become logical operators, i.e., they lie in the normalizer but not in the stabilizer. As a result, the minimum distance was bounded by the row weight. This is a structural limitation we want to avoid. **[Next slide.]**

Slide 4: Design Principle (about 1:20)

The core idea is simple: enforce orthogonality only on the active part. We require $H_X H_Z^{'} = 0$, but we intentionally keep the latent part non-orthogonal. This prevents small-weight removed rows from becoming logicals. We define latent-based distances as the minimum weights of logical operators supported on the latent-row spans (for X and Z). These distances isolate logicals induced by the latent rows, so they directly measure the risk created by row removal. Then we design the construction to make these distances large. **[Next slide.]**

Slide 5: Generalized Hagiwara–Imai Construction (Example J=3, L=12) (about 2:10)

We now present the construction for the proposed codes. First, J and L denote the column and row weights. We prepare $L/2$ permutation matrices F_i and G_j of size P. In the experiments shown later, we take $P = 768$. We build $\hat{H}_X, \hat{H}_Z$ by arranging F_i and G_j so that both the left and right halves are block-circulant. We use the top J block rows of $\hat{H}_X, \hat{H}_Z$ as the active matrices, and the remaining block rows as the latent matrices. Here a prime $(^{'})$ denotes transpose, or equivalently the inverse for permutation blocks. This structure reduces the parent-product to a small set of blocks Ψ_r. Each Ψ_r is a sum of $L/2$ swapped-order pairs, in other words, commutator terms, so commutativity between F_i

and G_j controls when Ψ_r becomes zero. The key idea is to control commutativity between F_i and G_j so that the top-left block is zero, i.e., the active blocks are orthogonal, while the off-diagonal blocks are nonzero, i.e., the active–latent products are not. On the next slide, we show how to enforce its orthogonality. **[Next slide.]**

Slide 6: Setting Δ and Γ (about 1:30)

Look at the parent-product on the left. In the top-left 3×3 active block, Ψ_3 does not appear. Recall that each Ψ_r is a sum of commutator terms. On the right, I color the (i, j) pairs in Γ by the same Ψ_r colors; these are exactly the pairs that must commute to force the corresponding Ψ_r to zero. Therefore, to make the active product zero, it needs to enforce commutativity on Γ. Pairs outside Γ are free to choose. To avoid the case where everything commutes, we need $4J \leq L$. This means the quantum coding rate is at least $1/2$. **[Next slide.]**

Slide 7: Affine Permutation Matrices (about 1:20)

Here is the search story. We start with three requirements: commute on Γ, keep at least one non-commuting pair outside Γ, and avoid short cycles in the active Tanner graphs. An exhaustive search over all permutation matrices is combinatorial. So we restrict the search space to a structured family that makes the commutativity and cycle checks cheap. This turns the problem into a practical sequential construction. We have released the construction software on GitHub. **[Next slide.]**

Slide 8: Example: Proposed Code Construction (about 1:00)

Here is one concrete instance. With $P = 768$, the construction gives a girth-8, $(3, 12)$-regular code with parameters $[[9216, 4612, \leq 48]]$. Earlier constructions were capped at distance 12 by row-weight bounds, but in this instance we find no logicals achieving that bound. We can explicitly build weight-48 logical operators, so $d_{\min} \leq 48$ is an upper bound. For this instance, we can prove $d_X^{(\mathrm{lat})} = d_Z^{(\mathrm{lat})} = 48$; I omit the proof here. To prove $d_{\min} = 48$, we would also need to rule out small-weight operators in the remaining range. We cannot rule out smaller logicals yet, but in deep error-floor experiments we saw no logical failures, so we expect the true distance is close to 48. **[Next slide.]**

Slide 9: Decoding Algorithm (BP + Post-Processing) (about 1:20)

Let me describe the decoder. First, we run joint BP on H_X and H_Z, exploiting X/Z correlations. Most frames are corrected by BP alone, but occasionally BP stalls with a small residual. When the number of unsatisfied checks is small (e.g., ≤ 10), we invoke post-processing. In post-processing, we estimate a set of suspicious bits K using OSD, flip-history, or a small library of elementary trapping sets, and form the residual syndrome by removing the complement contribution. If the solution space is low-dimensional and yields a small-weight update, we apply it; otherwise we keep the BP output. We have released the decoder software on GitHub. **[Next slide.]**

Slide 10: Performance Highlight (about 1:20)

Let me highlight the main result. We constructed a girth-8, (3,12)-regular code with these parameters. With joint BP and post-processing, the FER reaches 10^{-8} over depolarizing channels with physical error rate $p = 4\%$. At $p = 0.04$, we also begin to see early signs of an error floor. However, the remaining decoding failures leave only a small number of unsatisfied checks, which suggests they may be correctable by expanding the ETS library. Most importantly, the observed BP performance closely matches the DE benchmark. Here the DE benchmark is computed for a random non-orthogonal (3,12) code under the cycle-free assumption. This figure shows the corresponding FER curve. **[Next slide.]**

Slide 11: Conclusion (about 1:00)

To summarize: active-only orthogonality avoids the distance penalty. APMs give direct control of commutativity and short cycles. We constructed a girth-8, $(3, 12)$-regular code with strong BP performance. Thank you for your attention. I would be happy to take questions. **[End.]**